ONLINE LEARNING-GUIDED LEARNING RATE ADAPTATION VIA GRADIENT ALIGNMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

The performance of an optimizer on large-scale deep learning models depends critically on *fine-tuning* the learning rate, often requiring an extensive grid search over base learning rates, schedules, and other hyperparameters. We propose GALA (*Gradient Alignment-based Learning rate Adaptation*), a theory-inspired rule that dynamically adjusts the learning rate by tracking the alignment between consecutive gradients and using a local curvature estimate. Guided by the convergence analysis, we formulate the problem of selecting the learning rate as a one-dimensional online learning problem. When paired with an online learning algorithm such as Follow-the-Regularized-Leader, our method produces a flexible, adaptive learning rate schedule that tends to increase when consecutive gradients are aligned and decrease otherwise. We establish a data-adaptive convergence rate for normalized SGD equipped with GALA in the smooth, nonconvex setting. Empirically, common optimizers such as SGD and Adam, when augmented with GALA, demonstrate robust performance across a wide range of initial learning rates and perform competitively without the need for tuning.

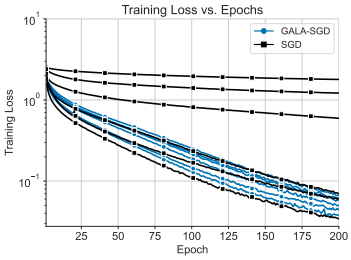
1 INTRODUCTION

Stochastic first-order (SFO) methods such as SGD (Robbins & Monro, 1951), AdaGrad (McMahan & Streeter, 2010; Duchi et al., 2011), and Adam (Kingma & Ba, 2015) have been the workhorse for training large-scale models due to their low computational overhead and strong empirical performance. Essentially, the practical performance of SFO methods relies on two components: the choice of the base learning rate and how the learning rate evolves during training. The initial selection is typically done by running a grid search over a range of values, which is referred to as *tuning*. On top of that, the evolution of the learning rate is most commonly controlled by scaling it externally via a *scheduler*. Depending on the characteristics of the optimizer, the learning rate could also be dynamically updated by some internal mechanism during training. For instance, SGD is often run with a *constant* learning rate and coupled with a scheduler such as cosine annealing (Loshchilov & Hutter, 2017) or linear decay (Defazio et al., 2023) that guides the learning rate following a *predetermined* rule. Similarly, the adaptive methods update the learning rate internally by accumulating observed gradients based on a rule that usually decays the learning rate below its initial value. Although optimizers have other parameters such as momentum and weight decay, they are often fixed at the beginning, whereas the learning rate evolves throughout and thus has a larger impact on the final performance.

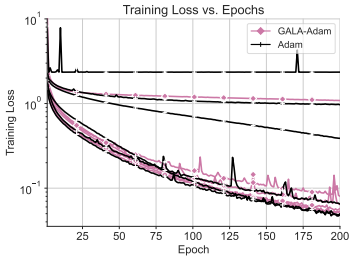
However, it is unclear how to choose an "empirically viable" combination of a base learning rate, optimizer, and scheduler, *a priori*, without tuning over a manually chosen set of parameters. In practice, this issue is compounded by the fact that hyperparameters are often tuned for only a few epochs or on a smaller-scale model, and then directly transferred to large-scale experiments. Yet, it has been documented that early training behaviors can be misleading (Wen et al., 2025) as such configurations will be far from ideal at the true scale. Therefore, this challenge highlights the need for a theoretically principled approach to learning rate adaptation that is robust, flexible, and provable. To put things in perspective, we highlight three key drawbacks of common approaches in this domain.

Robustness: Many optimizers, such as SGD and AdaGrad, are highly sensitive to the initial learning rate: an excessively large value can lead to divergence, while a very small one results in stagnation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) SGD versus SGD-GALA



(b) Adam versus ADAM-GALA

Figure 1: Training loss for standalone SGD and Adam versus GALA applied on their learning rates (SGD-GALA and ADAM-GALA) for initial values $[1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}]$.

Ideally, we seek stable performance across a wide range of initial learning rates, allowing for robust training even with suboptimal initializations.

Adaptivity: In most cases, the learning rate tends to decay over time, via internal dynamics or an external scheduler, limiting flexibility. For instance, the AdaGrad algorithm reduces the learning rate below its initial value, while many schedulers induce a decaying behavior for the learning rate on top of the internal mechanics. A desirable alternative would be a principled and adaptive scheduling mechanism capable of both increasing and decreasing the learning rate as needed.

Principled: Most theoretical frameworks in this area are grounded in convex optimization. For instance, AdaGrad and its variants are supported by data-dependent regret bounds derived from online convex optimization. However, the decaying nature of their learning rate is not empirically favorable for non-convex landscapes. We seek a learning rate rule whose design is derived from a non-convex analysis, with a provable guarantee for at least one principled instantiation.

Our contributions: While several existing works partially address these three limitations, none meets all the desired criteria simultaneously. To this end, we propose **G**radient **A**lignment-based **L**earning rate **A**daptation (GALA), a theory-inspired, online-learning guided framework for adjusting the learning rate on-the-fly by monitoring the evolution of the optimization path. Our goal with designing this framework is to **make algorithms robust to parameter-tuning** while **maintaining competitive performance**. In particular,

1. We construct a one-dimensional online loss function for selecting the learning rate, using the *alignment between stochastic gradients* and a *local estimate of the gradient Lipschitz constant*. The learning rate is then updated by solving the online problem using a suitable algorithm.
2. Our approach enables dynamic learning rate adaptation: it tends to *increase* when the gradients are aligned and *decrease* when misalignment is detected. We carefully moderate the alignment signal using a regularization term based on a local gradient Lipschitz estimate for stability.
3. We provide a regret-based analysis and establish convergence guarantees for one principled instantiation: normalized SGD with momentum using independent samples (Section 4).
4. Our method, combined with practical heuristics, shows strong, stable empirical performance for a wide range of initial learning rates. We propose heuristic implementations for other SFO methods.

Figure 1 provides a glimpse at the performance of our framework when applied to SGD and Adam. We show that GALA helps mitigate sensitivity to the initialization of the learning rate while maintaining competitive performance with respect to the best-performing runs of the standalone SGD and Adam.

1.1 RELATED WORK

In this section, we provide a concise literature review, the full version of which is found in Section A.

Stochastic first-order methods. Dating back to the seminal work Robbins & Monro (1951), the theoretical behavior of SGD and its many variants have been extensively studied. Considering general smooth functions, it is well-known that the learning rate must decrease at a rate of $\eta_t = O(1/\sqrt{t})$

where t is the iteration counter, and must also satisfy $\eta_t \leq O(1/L)$. Ghadimi & Lan (2013) established that SGD with a properly chosen learning rate achieves a complexity of $\mathcal{O}(\epsilon^{-2} + \sigma^2\epsilon^{-4})$, interpolating between deterministic and stochastic rates and matching the lower bounds (Arjevani et al., 2023). However, the choice of the learning rate depends on the problem parameters, e.g., L and σ, which are typically unknown or difficult to estimate in practice. Similar requirements are in place for the learning rate when the loss is ρ-weakly convex (Davis et al., 2020).

Adaptive and parameter-free optimization methods. AdaGrad (McMahan & Streeter, 2010; Duchi et al., 2011) was introduced for minimizing a sequence of *online* convex losses. It uses a time-varying learning rate based on accumulated squared norms of stochastic gradients. This idea paved the way for many algorithms, e.g., Adam (Kingma & Ba, 2015), RMSProp Tieleman & Hinton (2012), Adadelta Zeiler (2012), and their variants, which demonstrate strong empirical performance. Convergence guarantees have been established for such adaptive methods in both convex (Levy, 2017; Levy et al., 2018; Kavis et al., 2019; Joulani et al., 2020; Antonakopoulos et al., 2022; Liu et al., 2023a; Rodomanov et al., 2024) and non-convex settings (Li & Orabona, 2019; Ward et al., 2020; Li & Orabona, 2020; Kavis et al., 2022; Gadat & Gavra, 2022; Faw et al., 2022; Attia & Koren, 2023; Liu et al., 2023b), while showing that these guarantees can be achieved without prior knowledge of problem parameters such as smoothness and noise variance.

These methods are sensitive to the initial learning rate as they need to know an estimate of the initial distance to solution, $\|\boldsymbol{x}_0 - \boldsymbol{x}^*\|$. Recently, *parameter-free* optimization (Carmon & Hinder, 2022; Ivgi et al., 2023; Defazio & Mishchenko, 2023; Khaled et al., 2023; Mishchenko & Defazio, 2024; Kreisler et al., 2024; Attia & Koren, 2024) focuses on augmenting robustness to initialization by combining AdaGrad-type learning rate with a certificate that lower bounds $\|\boldsymbol{x}_0 - \boldsymbol{x}^*\|$. Although this enables the learning rate to increase, the scaling factor is practically bounded, restricting flexibility. Malitsky & Mishchenko (2020; 2024); Li & Lan (2025) studied parameter-free methods that estimate the time evolution of local smoothness for convex, deterministic problems. They differ from prior methods in having a non-monotone learning rate. A downside of these methods is empirical instability; if the increasing behavior is not tamed properly, optimization performance could be unstable.

Online learning-guided methods. Focusing on parameter-free online learning (Orabona & Pál, 2016), Orabona & Tommasi (2017) reformulated SGD as coin-betting game and applied a betting algorithm to eliminate the need to manually tune the learning rate for convex and quasi-convex objectives. Cutkosky et al. (2023a) proposed a technique for adaptively scaling any base optimizer and learning rate schedule, which is grounded in a black-box reduction framework Cutkosky & Orabona (2018). Zhuang et al. (2019) studied non-convex stochastic optimization and designed a surrogate loss technique for learning rate update. However, their method requires knowledge of problem-dependent parameters, e.g., gradient Lipschitz constant, thereby limiting flexibility.

2 PRELIMINARIES

We consider the stochastic optimization problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} \; F(\boldsymbol{x}) = \mathbb{E}_{\xi\sim\mathcal{D}}[f(\boldsymbol{x};\xi)],$$

where $f(\cdot;\xi)$ is a random function indexed by a random variable ξ drawn from distribution $\mathcal{D}$. The objective function $F : \mathbb{R}^d \to \mathbb{R}$ is assumed to be differentiable, possibly nonconvex and bounded from below, i.e., $F(\boldsymbol{x}) > -\infty$. Moreover, we make the following assumptions.

Assumption 2.1. Gradient noise has bounded variance: $\mathbb{E}[\|\nabla F(\boldsymbol{x}) - \nabla f(\boldsymbol{x};\xi)\|^2] \leq \sigma^2, \forall \boldsymbol{x} \in \mathbb{R}^d$.

Assumption 2.2. The gradient of $f(\boldsymbol{x};\xi)$ is L-Lipschitz continuous in $\boldsymbol{x}$, i.e., $\|\nabla f(\boldsymbol{x};\xi) - \nabla f(\boldsymbol{y};\xi)\| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|$ for all ξ and all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$.

Remark 2.3. Assumption 2.2 is slightly stronger than what is required for the analysis. Our convergence bound continues to hold when the above inequality applies *locally* along the optimization trajectory. We state the assumption in this global form for simplicity.

Background on online learning: In online learning, a learner makes decisions iteratively over rounds. At each round $t = 1, \cdots, T$: (i) Learner selects $\boldsymbol{x}_t \in \mathcal{X}$ from a set of actions; (ii) Environment reveals the loss function $\ell_t(\cdot)$; (iii) Learner suffers the loss $\ell_t(\boldsymbol{x}_t)$. The learner chooses its action $\boldsymbol{x}_t$ in round t *prior to* observing the loss $\ell_t(\cdot)$. The performance of the learner is measured by *regret*,

which is defined as the cumulative loss of the learner compared against a fixed action $\boldsymbol{x}$:

$$\mathrm{Reg}_T(\boldsymbol{x}) = \sum_{t=1}^{T} (\ell_t(\boldsymbol{x}_t) - \ell_t(\boldsymbol{x})). \tag{1}$$

The goal is to achieve *sublinear regret*, i.e., $\mathrm{Reg}_T(\boldsymbol{x}) = o(T)$, such that the time average of regret goes to zero as $T \to \infty$, and the learner performs as well as the fixed strategy in the limit.

3 Online learning rate selection

We begin by introducing a simplified template that outlines our design. Our primary goal is to provide insight into the idea of gradient alignment, explain our adaptive strategy, and establish the foundation for the online learning formulation of the learning rate. Consider the SGD update rule

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \eta_t \boldsymbol{g}_t(\boldsymbol{x}_t), \quad \boldsymbol{g}_t(\boldsymbol{x}_t) = \nabla f(\boldsymbol{x}_t; \xi_t), \tag{2}$$

where $\xi_t \sim \mathcal{D}$ is a random sample drawn from the distribution $\mathcal{D}$ at iteration t. Our goal is to choose a sequence of learning rates guided by the progress of the algorithm, as measured by the function value difference defined by $\Delta_{t+1} = F(\boldsymbol{x}_{t+1}) - F(\boldsymbol{x}_t)$. At this point, we deviate from the classical analysis; inspired by Cutkosky et al. (2023b), we apply the fundamental theorem of calculus to get

$$\Delta_{t+1} = \langle \boldsymbol{\nabla}_t, \boldsymbol{x}_{t+1} - \boldsymbol{x}_t \rangle = -\eta_t \langle \boldsymbol{\nabla}_t, \boldsymbol{g}_t(\boldsymbol{x}_t) \rangle, \tag{3}$$

where $\boldsymbol{\nabla}_t = \int_0^1 \nabla F(\boldsymbol{x}_t + \lambda(\boldsymbol{x}_{t+1} - \boldsymbol{x}_t))\, d\lambda$ denotes the average gradient along the line segment between $\boldsymbol{x}_t$ and $\boldsymbol{x}_{t+1}$. Note that the right-hand side of Eq. 3 concerns the *alignment* between the gradients $\boldsymbol{\nabla}_t$ and $\boldsymbol{g}_t(\boldsymbol{x}_t)$ and serves as a useful signal for adjusting the learning rate. When the inner product $\langle \boldsymbol{\nabla}_t, \boldsymbol{g}_t(\boldsymbol{x}_t) \rangle$ is positive, it indicates that the gradients point in similar directions and increasing the learning rate may lead to greater progress. Conversely, a negative alignment implies opposing directions, in which case a smaller learning rate may be more appropriate.

However, computing $\boldsymbol{\nabla}_t$ is generally intractable, as it involves the true gradient and an integral. A key observation in Zhang et al. (2020); Cutkosky et al. (2023b) is that an unbiased estimate of $\boldsymbol{\nabla}_t$ can be constructed by evaluating the gradient at a random point along the line segment. Specifically, let λ_t be a random variable uniformly distributed over $[0, 1]$, and let ξ_t' be an independent sample from the distribution $\mathcal{D}$. Then for $\boldsymbol{w}_t = \boldsymbol{x}_t + \lambda_t(\boldsymbol{x}_{t+1} - \boldsymbol{x}_t)$ and $\boldsymbol{g}_t'(\boldsymbol{w}_t) = \nabla f(\boldsymbol{w}_t; \xi_t')$, we have $\boldsymbol{\nabla}_t = \mathbb{E}_{\lambda_t}[\nabla F(\boldsymbol{w}_t)] = \mathbb{E}_{\lambda_t, \xi_t'}[\boldsymbol{g}_t'(\boldsymbol{w}_t)]$, which implies

$$F(\boldsymbol{x}_{t+1}) - F(\boldsymbol{x}_t) = -\eta_t \mathbb{E}_{\lambda_t, \xi_t'}[\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t) \rangle]. \tag{4}$$

To maximize the function value decrease, Eq. (4) suggests that a natural objective is to maximize $\eta_t \mathbb{E}_{\lambda_t, \xi_t'}[\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t) \rangle]$, which presents two challenges. The first one is related to the convergence metric. This approach only leads to an upper bound on $\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}[\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t) \rangle]$, and does not provide a meaningful bound on the gradient norm of F, which is the standard metric one seeks to obtain. Our idea is to decompose the inner product as $\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t) \rangle = \langle \boldsymbol{g}_t'(\boldsymbol{x}_t), \boldsymbol{g}_t(\boldsymbol{x}_t) \rangle + \langle \boldsymbol{g}_t'(\boldsymbol{w}_t) - \boldsymbol{g}_t'(\boldsymbol{x}_t), \boldsymbol{g}_t(\boldsymbol{x}_t) \rangle$, where $\boldsymbol{g}_t'(\boldsymbol{x}_t) = \nabla f(\boldsymbol{x}_t; \xi_t')$. Then, the first term leads to $\mathbb{E}[\langle \boldsymbol{g}_t'(\boldsymbol{x}_t), \boldsymbol{g}_t(\boldsymbol{x}_t) \rangle] = \mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|^2]$ and the second can be controlled by the Lipschitz constant of the gradient.

The second issue is that the minimization of the right-hand side of Eq. (4) with respect to the learning rate η_t is an *implicit* problem. The objective depends on the interpolated point $\boldsymbol{w}_t$, which could be determined only *after* the learning rate η_t is chosen. The solution is to cast the learning rate selection as an *online learning problem*, and derive a sequence of online loss functions that will govern the selection process. We combine and formalize both ideas in the following lemma.

Lemma 3.1. *Define the gradient Lipschitz estimate* $L_t = \frac{\|\boldsymbol{g}_t'(\boldsymbol{w}_t) - \boldsymbol{g}_t'(\boldsymbol{x}_t)\|}{\|\boldsymbol{w}_t - \boldsymbol{x}_t\|}$ *and the surrogate loss*

$$\ell_t(\eta) \triangleq -\eta \langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t) \rangle + 0.5 L_t \|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2 \eta^2. \tag{5}$$

Suppose that Assumption 2.1 holds and $L_t \le L^{\max}$ *for all* $t \ge 0$ *with probability one. Then* $\sum_{t=0}^{T-1} \mathbb{E}[(\eta - \eta^2 L^{\max}) \|\nabla F(\boldsymbol{x}_t)\|^2] \le \mathbb{E}[F(\boldsymbol{x}_0) - F(\boldsymbol{x}_T) + L^{\max} T \eta^2 \sigma^2 + \sum_{t=0}^{T-1} (\ell_t(\eta_t) - \ell_t(\eta))]$.

Our surrogate loss function $\ell_t(\eta)$ in Eq. 5 consists of two terms. The first measures the *alignment* between two consecutive (stochastic) gradients $\boldsymbol{g}_t'(\boldsymbol{w}_t)$ and $\boldsymbol{g}_t(\boldsymbol{x}_t)$, and the second is a quadratic regularization term that depends on the local Lipschitz constant estimate L_t. The online nature of

Algorithm 1 SGD-GALA

Require: Initial point $\boldsymbol{x}_0$, initial learning rate η_0, maximum learning rate $\eta^{\max}$, $\delta > 0$

1: **for** $t = 0, 1, \ldots, T$ **do**
2: Sample $\xi_t \sim \mathcal{D}$ and compute $\boldsymbol{g}_t(\boldsymbol{x}_t) = \nabla f(\boldsymbol{x}_t; \xi_t)$
3: $\boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}_t - \eta_t \boldsymbol{g}_t(\boldsymbol{x}_t)$
4: Sample $\xi_t' \sim \mathcal{D}$ and compute $\boldsymbol{g}_t'(\boldsymbol{x}_t) = \nabla f(\boldsymbol{x}_t; \xi_t')$
5: Sample $\boldsymbol{s}_t \sim \mathrm{Uniform}[0, 1]$ and compute $\boldsymbol{w}_t \leftarrow \boldsymbol{x}_t + s_t(\boldsymbol{x}_{t+1} - \boldsymbol{x}_t)$ and $\boldsymbol{g}_t'(\boldsymbol{w}_t) = \nabla f(\boldsymbol{w}_t; \xi_t')$
6: $L_t \leftarrow \|\boldsymbol{g}_t'(\boldsymbol{w}_t) - \boldsymbol{g}_t'(\boldsymbol{x}_t)\| / \|\boldsymbol{w}_t - \boldsymbol{x}_t\|$
7: $\eta_{t+1} \leftarrow \mathrm{clip}_{[0,\eta^{\max}]}\left(\dfrac{\sum_{s=0}^{t}\langle \boldsymbol{g}_s'(\boldsymbol{w}_s), \boldsymbol{g}_s(\boldsymbol{x}_s)\rangle}{\delta + \sum_{s=0}^{t} L_s \|\boldsymbol{g}_s(\boldsymbol{x}_s)\|^2}\right)$
8: **end for**

the problem is due to the fact that $\boldsymbol{g}_t'(\boldsymbol{w}_t)$ and L_t can only be computed after the learning rate η_t is chosen. Moreover, η in the inequality of Lemma 3.1 is the comparator of our online learning problem and can be chosen arbitrarily in our analysis. If we achieve a low regret for the online learning problem (see Section 4), then a proper choice of η will lead to a complexity of $\mathcal{O}(\epsilon^{-2} + \sigma^2\epsilon^{-4})$.

Our approach is inspired by both Cutkosky et al. (2023b) and Zhuang et al. (2019). Compared to Cutkosky et al. (2023b), the key difference is that their method applies online learning to guide the choice of the update direction, whereas we focus on selecting the learning rate. In contrast to Zhuang et al. (2019), our method differs in two major ways: (i) we estimate the local gradient Lipschitz constant on the fly, instead of relying on a global Lipschitz estimate; and (ii) for the first term, we use the alignment between two stochastic gradients evaluated at different points $\boldsymbol{w}_t$ and $\boldsymbol{x}_t$, while Zhuang et al. (2019) uses gradients at the same point. We summarize the key differences below:

	Zhuang et al. (2019)	GALA (Ours)
Alignment signal	Same iterate, indep. batches	Different iterates, same batch
Curvature	Global L (known a priori)	Local L_t (estimated online)
Analysis	Quadratic upper bound	Fundamental thm. of calculus
Required parameter	L	$\delta, \eta^{\max}$ in theory; none in practice

The next step is using an online learning algorithm that operates on the loss sequence ℓ_t to update the learning rate η_t. Since the loss functions are quadratic, we have several options to choose from. As an example, the Follow-the-Regularized-Leader (FTRL) algorithm is given by

$$\eta_{t+1} = \arg\min_{\eta \in [0,\eta^{\max}]} \left\{ \sum_{s=0}^{t} \ell_s(\eta) + 0.5\delta\eta^2 \right\},$$

where $\eta^{\max}$ is the maximal learning rate and $\delta \geq 0$ is a user-defined constant to ensure stability. Using the loss in Eq. 5 and the FTRL, we obtain a closed-form expression for η_{t+1}:

$$\eta_{t+1} = \mathrm{clip}_{[0,\eta^{\max}]}\left(\frac{\sum_{s=0}^{t}\langle \boldsymbol{g}_s'(\boldsymbol{w}_s), \boldsymbol{g}_s(\boldsymbol{x}_s)\rangle}{\delta + \sum_{s=0}^{t} L_s \|\boldsymbol{g}_s(\boldsymbol{x}_s)\|^2}\right), \tag{6}$$

where $\mathrm{clip}_{[0,\eta^{\max}]}(\cdot)$ clips a real-valued input to the interval $[0, \eta^{\max}]$. Note that the learning rate incorporates directional information via the alignment term along the optimization path: when the gradients are aligned, the learning rate is encouraged to increase; when they are misaligned, it decreases. Additionally, the quadratic term moderates the learning rate update based on the magnitude of observed gradients. This adaptive, inherent feature of our framework holds by default for various choices of online learners, such as OGD (Zinkevich, 2003; Orabona, 2019).

Remark 3.2. We compute two stochastic gradients at two points using the same batch to form a local Lipschitz estimate L_t, and update the learning rate as in Eq. 6. This doubles the gradient cost relative to SGD, as is standard for this class of parameter-free methods Malitsky & Mishchenko (2020); Ji & Lan (2026). This extra computation enables adaptivity to local curvature without manual tuning; if a tight upper bound on L_t were known, these evaluations would be redundant. Thus, we trade a constant-factor increase in per-step cost to eliminate the larger burden of hyperparameter search.

Remark 3.3. As stated in Alg. 1, our method originally evaluates three stochastic gradients per step. To reduce the cost, we reuse stochastic gradients by setting $\xi_{t+1} = \xi_t'$ and $s_t = 1$ such that

the gradient $\nabla f(\boldsymbol{w}_t; \xi_t')$ coincides with $\nabla f(\boldsymbol{x}_{t+1}; \xi_{t+1})$. This reduces the gradient cost to two per iteration (see Eq. 10, Section 5). The modification $\xi_{t+1} = \xi_t'$ is compatible with our analysis, since it does not require independence between ξ_{t+1} and ξ_t'. Replacing the random variable s_t with the deterministic choice $s_t = 1$ is an approximation with negligible impact on performance. Please see Section E.2.5 for empirical validation of these practical considerations.

4 Convergence analysis

In this section, we analyze a variant of our proposed method in Algorithm 1. Instead of using the standard SGD update in Eq. 2, we instantiate the GALA framework with normalized SGD with momentum (Cutkosky & Mehta, 2020) and establish its convergence rate for stochastic nonconvex optimization. The main theoretical advantage of using a normalized update is that it simplifies the surrogate loss function, making the regret bound easier to establish. The core online learning principle—aligning consecutive gradients and regularizing with local curvature—is shared across all instantiations of our framework. Specifically, we consider the update rule

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - (\eta_t/\|\boldsymbol{m}_t\|)\boldsymbol{m}_t, \quad \boldsymbol{m}_t = (1-\alpha)\boldsymbol{m}_{t-1} + \alpha \nabla f(\boldsymbol{x}_t; \xi_t), \tag{7}$$

where $\alpha \in (0, 1]$ is the momentum parameter. The normalization ensures that the learning rate η_t directly controls the distance between $\boldsymbol{x}_{t+1}$ and $\boldsymbol{x}_t$, thereby promoting stability. However, it could also amplify the gradient noise. To mitigate this and reduce the variance, we apply an exponential moving average in Eq. 7, which is governed by the momentum parameter α. To accommodate the different update rule in Eq. 7, the surrogate loss function must be modified accordingly as:

$$\ell_t^{\mathrm{N}}(\eta) = -\eta \big\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \tfrac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|} \big\rangle + \big(\tfrac{\max\{L_t, M\}}{2} + \tfrac{4(1-\alpha)\tilde{L}_t}{3\alpha}\big)\eta^2, \tag{8}$$

where $M > 0$ is a user-defined constant and $\tilde{L}_t = \frac{\|\boldsymbol{g}_t'(\boldsymbol{x}_{t+1}) - \boldsymbol{g}_t'(\boldsymbol{x}_t)\|}{\|\boldsymbol{x}_{t+1} - \boldsymbol{x}_t\|}$ is a second local gradient Lipschitz estimate. There are three main differences compared to Eq. 5. First, the linear term in Eq. 8 measures the alignment between the gradient $\boldsymbol{g}_t'(\boldsymbol{w}_t)$ and the normalized update direction $\frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}$, rather than with the stochastic gradient $\boldsymbol{g}_t(\boldsymbol{x}_t)$. In addition, the quadratic term is independent of the norm of the stochastic gradient $\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|$ due to the normalization. Moreover, it includes an additional regularization term that depends on the momentum parameter α and the local gradient Lipschitz estimate $\tilde{L}_t$, which arises from the analysis of momentum. In the following theorem, we establish the convergence rate of the update rule in Eq. 7 in terms of the regret with respect to the new surrogate loss in Eq. 8 (see Appendix C for its proof).

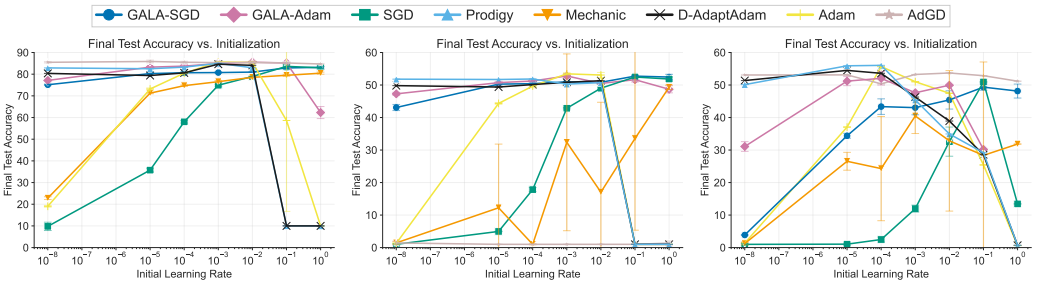
Remark 4.1. The quantity M is introduced to ensure that the surrogate loss in Eq. 8 is M-strongly convex and to facilitate the regret analysis in Lemma 4.3. This is also a design choice to ensure numerical stability in a similar spirit to the δ parameter in the learning rate of Adam.

Theorem 4.2. *Let $\{\boldsymbol{x}_t\}_{t=0}^{T-1}$ be generated by the update rule in Eq. 7 and $\eta_t \le \eta^{\max}$ for all t. Recall that $L_t = \frac{\|\boldsymbol{g}_t'(\boldsymbol{w}_t) - \boldsymbol{g}_t'(\boldsymbol{x}_t)\|}{\|\boldsymbol{w}_t - \boldsymbol{x}_t\|}$, $\tilde{L}_t = \frac{\|\boldsymbol{g}_t'(\boldsymbol{x}_{t+1}) - \boldsymbol{g}_t'(\boldsymbol{x}_t)\|}{\|\boldsymbol{x}_{t+1} - \boldsymbol{x}_t\|}$, and the surrogate loss $\ell_t^{\mathrm{N}}(\eta)$ defined in Eq. 8. Moreover, define the regret $\mathrm{Reg}_T^{\mathrm{N}} \triangleq \max_{\eta \in [0, \eta^{\max}]} \sum_{t=0}^{T-1} (\ell_t^{\mathrm{N}}(\eta_t) - \ell_t^{\mathrm{N}}(\eta))$, the initial function value gap $\Delta_F \triangleq F(\boldsymbol{x}_0) - F(\boldsymbol{x}^*)$, and the average Lipschitz estimate $L_T^{\mathrm{avg}} = \max\big\{\mathbb{E}\big[\frac{1}{T}\sum_{t=0}^{T-1} \max\{L_t, M\}\big], \mathbb{E}\big[\frac{1}{T}\sum_{t=0}^{T-1} \tilde{L}_t\big]\big\}$. For $\alpha = \min\{\frac{1}{\sigma\sqrt{T}}, 1\}$, it holds that*

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|] = \mathcal{O}\big(\tfrac{\sigma^{1/2}(L_T^{\mathrm{avg}}(\Delta_F + \mathbb{E}[\mathrm{Reg}_T^{\mathrm{N}}]))^{1/2}}{T^{1/4}} + \tfrac{\sigma^2}{\sqrt{T}} + \tfrac{\sqrt{L_T^{\mathrm{avg}}(\Delta_F + \mathbb{E}[\mathrm{Reg}_T^{\mathrm{N}}])}}{\sqrt{T}} + \tfrac{\Delta_F + \mathbb{E}[\mathrm{Reg}_T^{\mathrm{N}}]}{\eta^{\max} T}\big).$$

In Theorem 4.2, the choice of α is made to obtain the best dependence on both T and σ. We note that assuming knowledge of T is standard in the literature and typically not restrictive in practice. If σ is unknown, one could choose $\alpha = 1/\sqrt{T}$ and still obtain the same convergence rate, albeit with a worse dependence on σ. Moreover, the rate in Theorem 4.2 depends on the regret of the associated online learning problem. Hence, we propose to use an online algorithm to adaptively update the learning rate η_t. Since the loss function in Eq. 8 is strongly convex, a logarithmic regret is possible. To obtain the best theoretical guarantees, we use optimistic FTRL (Rakhlin & Sridharan, 2013; Steinhardt & Liang, 2014; Mohri & Yang, 2016):

$$\eta_{t+1} = \arg\min_{\eta \in [0, \eta^{\max}]} \big\{\textstyle\sum_{s=0}^{t} \ell_s^{\mathrm{N}}(\eta) + \frac{\delta}{2}\eta^2 + h_{t+1}(\eta)\big\}, \tag{9}$$



(a) **CIFAR-10** (b) **CIFAR-100** (c) **Flower102**

Figure 2: **Final test accuracy** for training **ResNet-18** on CIFAR-10, CIFAR-100 and Flower102 from different initial learning rates.

where $h_{t+1}(\cdot)$ is a hint function that aims to approximate the next loss function ℓ^{N}_{t+1}. Specifically, note that $\boldsymbol{m}_{t+1}$ is already known at the time when we select η_{t+1}. Hence, we set $h_{t+1}(\eta) = -\eta\langle \boldsymbol{g}_{t+1}(\boldsymbol{x}_{t+1}), \frac{\boldsymbol{m}_{t+1}}{\|\boldsymbol{m}_{t+1}\|}\rangle$, which yields the following closed-form update rule:

$$\eta_{t+1} = \mathrm{clip}_{[0,\eta^{\max}]}\Big(\frac{\sum_{s=0}^{t}\langle \boldsymbol{g}'_s(\boldsymbol{w}_s), \boldsymbol{m}_s/\|\boldsymbol{m}_s\|\rangle + \langle \boldsymbol{g}_{t+1}(\boldsymbol{x}_{t+1}), \boldsymbol{m}_{t+1}/\|\boldsymbol{m}_{t+1}\|\rangle}{\delta + \sum_{s=0}^{t}(\max\{L_s, M\} + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_s)}\Big).$$

We bound the regret of the above update rule in the following lemma (see Appendix D for the proof).

Lemma 4.3. *Let* $\eta^{\max} = \sqrt{\alpha}\bar{\eta}$ *for some given* $\bar{\eta}$. *Suppose that* $\max_{t\in[T]}\{L_t, \tilde{L}_t\} \le L^{\max}$. *Then we have* $\mathbb{E}[\mathrm{Reg}^{\mathrm{N}}_T] = \mathcal{O}\big(\bar{\eta}^2 \max\{L^{\max}, M\}\log\big(1 + \frac{\max\{L^{\max}, M\}}{\alpha\delta}T\big) + \frac{\sigma^2}{M}\log T\big)$.

Remark 4.4. Note that the convergence rate in Theorem 4.2 and the regret bound in Lemma 4.3 do not explicitly depend on the initial learning rate η_0; instead, η_0 influences the bound indirectly via the initial gradient Lipschitz estimates L_0, $\tilde{L}_0$. As a result, our method is robust to a wide range of initializations, as demonstrated in the experiments in Section 5. In contrast, the convergence rate of SGD has the form $\frac{\Delta_F + \eta_0^2\sigma^2}{\eta_0\sqrt{T}}$, so when η_0 is too small/large, it *multiplicatively* slows down the rate.

Combining Thm. 4.2 and Lem. 4.3, we show that our method achieves a convergence rate of $\mathcal{O}(\frac{\sigma^{1/2}}{T^{1/4}} + \frac{1}{\sqrt{T}})$ up to logarithmic factors, which matches Cutkosky & Mehta (2020) with a constant learning rate. Moreover, the convergence rate in Thm. 4.2 is in terms of the *trajectory-dependent* Lipschitz estimates, which can be much smaller than the global constant L used in Cutkosky & Mehta (2020). Notably, we achieve this by adaptively selecting the learning rate instead of using a predefined constant. Finally, we remark that our convergence results are comparable to those for AdaGrad (Faw et al., 2022; Attia & Koren, 2023; Liu et al., 2023b), with the key distinction that our learning rate can *both increase and decrease*, while that of AdaGrad is monotonically decreasing.

Remark 4.5. Our rate matches the $\Omega(\epsilon^{-4})$ lower bound for the standard stochastic first-order oracle (Arjevani et al., 2023). However, our analysis additionally relies on Assumption 2.2, under which variance-reduced methods attain the optimal $\mathcal{O}(\epsilon^{-3})$ complexity (Arjevani et al., 2023). Hence, our rate is not optimal in this setting; combining GALA with variance reduction is left for future work.

Remark 4.6. The parameter $\eta^{\max}$ is introduced primarily for theoretical purposes, allowing us to apply FTRL and its standard regret guarantee. In practice, we remove the clipping step in Eq. (6), and hence $\eta^{\max}$ is *not required* as a hyperparameter in our experiments.

5 Numerical Experiments

We present our results of applying GALA to training a residual network (He et al., 2016) on the CIFAR-10, CIFAR-100 (Krizhevsky, 2009) and Flower102 (Nilsback & Zisserman, 2008) as well as a Vision Transformer (Dosovitskiy et al., 2021) on Tiny-ImageNet (Stanford CS231n, 2015).

5.1 Implementation Details

SGD + GALA. When applying GALA to SGD, we introduce the following modifications to Alg. 1:

1. Instead of sampling a point $\boldsymbol{w}_t$ from the segment (cf. Line 5), we set $\boldsymbol{w}_t = \boldsymbol{x}_{t+1}$ (Remark 3.3).
2. To evaluate the alignment at time t, we compute both gradients *with the same mini-batch* ξ_{t+1}; i.e., we use $\langle \nabla f(\boldsymbol{x}_{t+1}; \xi_{t+1}), \nabla f(\boldsymbol{x}_t; \xi_{t+1}) \rangle$ as the first term of the surrogate loss in Eq. 5.
3. We omit learning rate clipping in Eq. (6), and eliminate the hyperparameter $\eta^{\max}$ (Remark 4.6).

The first and third modifications are mainly for simplicity, and the second is crucial for the empirical performance of GALA. A direct application of the theory would suggest computing the alignment $\langle \nabla f(\boldsymbol{x}_{t+1}; \xi_{t+1}), \nabla f(\boldsymbol{x}_t; \xi_t) \rangle$, where the two gradients are evaluated using independent mini-batches. In practice, however, due to gradient noise, this quantity tends to concentrate around zero, resulting in an overly small learning rate. To mitigate, we compute both gradients using the same mini-batch. This heuristic choice substantially improves practical performance; a rigorous theoretical justification is left for future work. Incorporating the changes, the learning rate update using FTRL becomes:

$$L_t = \frac{\|\nabla f(\boldsymbol{x}_{t+1}; \xi_{t+1}) - \nabla f(\boldsymbol{x}_t; \xi_{t+1})\|}{\|\boldsymbol{x}_{t+1} - \boldsymbol{x}_t\|}, \quad \eta_{t+1} = \frac{\sum_{s=0}^{t} \langle \nabla f(\boldsymbol{x}_{s+1}; \xi_{s+1}), \nabla f(\boldsymbol{x}_s; \xi_{s+1}) \rangle}{\sum_{s=0}^{t} L_s \|\boldsymbol{g}_s(\boldsymbol{x}_s)\|^2} \tag{10}$$

Adam + GALA. We adapt GALA to the Adam (Kingma & Ba, 2015) whose update rule is given by

$$\boldsymbol{m}_t = \beta_1 \boldsymbol{m}_{t-1} + (1-\beta_1)\nabla f(\boldsymbol{x}_t; \xi_t), \ \boldsymbol{v}_t = \beta_2 \boldsymbol{v}_{t-1} + (1-\beta_2)\nabla f(\boldsymbol{x}_t; \xi_t)^2,$$
$$\boldsymbol{d}_t = \boldsymbol{m}_t / \sqrt{\delta + \boldsymbol{v}_t}, \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \eta_t \boldsymbol{d}_t,$$

where all operations are element-wise, and we omit bias correction for simplicity. To select the learning rate η_t for Adam, we modify the surrogate loss in Eq. 5 as follows: (i) we replace the SGD direction $\nabla f(\boldsymbol{x}_t; \xi_t)$ with the Adam update direction $\boldsymbol{d}_t$; (ii) we substitute $\nabla f(\boldsymbol{w}_t; \xi_t')$ with $\nabla f(\boldsymbol{x}_t; \xi_t)$, so that the gradient alignment term involves the stochastic gradients computed on the same mini-batch ξ_t; (iii) we estimate L_t using the same heuristic in Eq. 10. These modifications lead to the following learning rate update:

$$\eta_{t+1} = \big(\sum_{s=0}^{t} \langle \boldsymbol{d}_s, \nabla f(\boldsymbol{x}_s; \xi_s) \rangle\big) / \big(\sum_{s=0}^{t} L_s \|\boldsymbol{d}_s\|^2\big).$$

5.2 Results on ResNet-18 and ViT-Tiny

Model and dataset. We use the `torchvision` implementation of the ResNet-18 model and train it on CIFAR-10, CIFAR-100 and Flower102. We apply random cropping and horizontal flipping during training. For ViT-Tiny, we use the `timm` implementation with pretrained weights and finetune it on Tiny-ImageNet-200. The training uses a richer set of augmentations, including random cropping, horizontal flipping, color jitter, random erasing, and the auto augmentation for ImageNet.

Optimizers. We apply GALA on SGD and Adam, denoted as SGD-GALA and Adam-GALA, respectively. For ResNet-18, we compare them against SGD, Adam, as well as parameter-free algorithms Mechanic (Cutkosky et al., 2023a), AdGD (Malitsky & Mishchenko, 2020), Prodigy (Mishchenko & Defazio, 2024) and D-Adaptation (Defazio & Mishchenko, 2023). For ViT-Tiny, we use SGD, AdamW, and Prodigy as baselines.

Setup. From the same model initialization, we run each method with initial learning rates from $[10^{-8}, 1]$ and fix all other parameters at their default values (see Appendix E). For ResNet-18, we run algorithms for 200 epochs with a training batch size of 128, a constant learning rate schedule and zero weight decay. For ViT-Tiny, we train for 100 epochs with a larger batch size to 1024 to maximize GPU utilization. We use a cosine schedule with weight decay of 0.05 for AdamW, SGD and Prodigy, while SGD-GALA and Adam-GALA are run with a constant schedule and zero weight decay.

Due to space limitations, we present a representative subset of our results, and an extensive empirical study with convergence and learning rate analysis is provided in Section E.

Results for ResNet-18. In Figure 2, we show the final test accuracy with respect to the initial learning rates. SGD-GALA and Adam-GALA perform consistently from a range of initial learning rates and are competitive with the best-performing method. We validate that GALA enables its base optimizer to achieve competitive performance for a *wider range of initial values, improving stability* for very large/small initializations. In comparison, AdGD has a small advantage in test accuracy on CIFAR-10 and Flower102 while it performs poorly on the CIFAR-100 dataset due to unstable

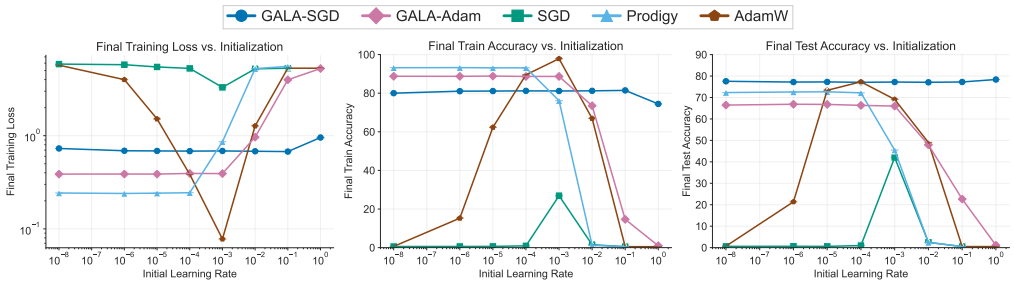


Figure 3: Training loss, training accuracy and test accuracy for training **ViT-Tiny** on **Tiny-ImageNet.**

learning rate evolution (see Figure 10c in Section E). Note that the official AdGD implementation requires a heuristic mechanism to limit the learning rate growth, introducing a new hyperparameter. Mechanic shows better performance for large initial values, which deteriorates noticeably as the learning rate gets smaller. Both Prodigy and D-Adaptation yield some of the best performances when the initial step size is small; but the performance drops sharply for initializations larger than 10^{-3}.

Results for ViT-Tiny. The robustness of GALA is more evident in this setup, as shown in Figure 3. SGD-GALA improves SGD across all initializations and achieves nearly identical performance for all values. ADAM-GALA shows a similar improvement, especially for initializations where AdamW performs poorly, although the best AdamW configuration still slightly outperforms ADAM-GALA. Notably, AdamW achieves its strongest performance only within a narrow range of learning rates (from 10^{-4} to 10^{-3}). Prodigy performs well for small learning rates but similarly degrades beyond 10^{-3}. Overall, SGD-GALA achieves *the best* test accuracy *across all initializations*.

Evolution of the learning rate. In Figure 4, we show how the learning rate evolves for ADAM-GALA. When initialized with a large learning rate, GALA tends to reduce it; this might be followed by an increasing regime or a sustained decrease depending on the base value. Conversely, it increases the learning rate when initialized with a small value, which enables it to recover from very small initializations. This behavior arises from the use of an online algorithm to select the learning rate, allowing it to adjust bidirectionally rather than following a fixed schedule. In the majority of cases, the learning rate for ADAM-GALA approaches the range between 10^{-4} and 10^{-3}, which corresponds to the best initial choices when running Adam (see Figure 2). This learning rate adaptation enables consistent convergence for methods coupled with GALA.

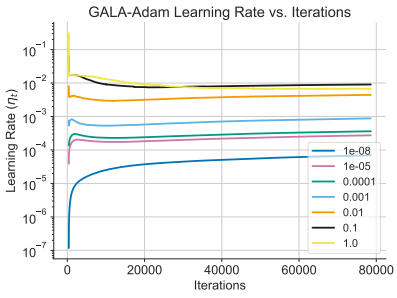


Figure 4: ADAM-GALA's learning rate evolution for training ResNet-18 on CIFAR-100.

Runtime vs. tuning trade-off. Although our implementation requires two gradient calls per iteration, its runtime is not doubled. In Table 1 (Appendix E.2.6), our timing logs on both ResNet and ViT-Tiny experiments suggest that GALA only adds about 50% overhead in runtime compared to standalone SGD and AdamW. We expect further improvements with a more optimized implementation, as our SGD baseline is the highly-optimized, native PyTorch implementation. On the other hand, our method promotes stable and competitive performance from a wide range of initializations, thereby substantially reducing the tuning cost.

6 CONCLUSION

We propose a theory-inspired framework, GALA, that dynamically adjusts the learning rate based on gradient alignment and a local curvature estimate. Motivated by convergence analysis, we formulate learning rate selection as a one-dimensional online learning problem and solve it using an online learning algorithm. We establish convergence guarantees for normalized SGD equipped with GALA and conduct preliminary experiments demonstrating that, when combined with SGD or Adam, our method yields robust performance across a wide range of initial learning rates.

AI USE STATEMENT

In this work, generative AI tools were used to support selected parts of the experimental implementation. In particular, the authors used AI assistance with setting up the experimental codebase, such as model and dataset loading, data augmentation, proper switching between training and evaluation modes, and interaction with cloud-based experiment tracking tools. They also assisted with retrieving and preprocessing recorded experimental results from experiment tracking workspaces. They were used to help generate and polish the figures.

The proposed algorithms were implemented by the authors where the generative AI tools were used to ensure that the optimizer implementations were compatible with the native PyTorch optimizer interface and conventions. Baseline methods were also implemented by the authors whenever they were unavailable in PyTorch or not released by the original authors. Generative AI tools helped with minor corrections to and verification of the implemented algorithms. All AI-assisted code was reviewed and verified by the authors.

Generative AI tools were additionally used for limited editorial support, including grammar correction, language polishing, and occasional refinement of phrasing for ideas and sentences already written by the authors.

Generative AI tools were not used to develop the main research ideas, theoretical framework, algorithmic design, mathematical claims, hypotheses, or proofs, all of which were developed solely by the authors. They were not used to interpret the experimental results.

The authors take full responsibility for the final content of the paper, including all text, claims, code, figures, and other artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work is primarily theoretical, and its empirical evaluation is conducted using standard experimental setups and benchmark datasets that are widely adopted in the field. Therefore, the authors are not aware of any ethical concerns specific to the theoretical and empirical contributions presented in this work beyond the broader ethical implications commonly recognized in machine learning research.

REPRODUCIBILITY STATEMENT

All assumptions are stated in Section 2, and complete proofs of Lemma 3.1, Theorem 4.2, and Lemma 4.3 are given in Appendices B to D.

The practical algorithms are specified in Section 5, and the models, datasets, data augmentation, baselines, hyperparameters, and hardware are described in Section 5 and Appendix E.

REFERENCES

Luís B Almeida, Thibault Langlois, José D Amaral, and Alexander Plakhov. Parameter adaptation in stochastic optimization. In *On-line learning in neural networks*, pp. 111–134. Cambridge University Press, 1999.

Kimon Antonakopoulos, Dong Quan Vu, Volkan Cevher, Kfir Yehuda Levy, and Panayotis Mertikopoulos. UnderGrad: A universal black-box optimization method with almost dimension-free convergence rate guarantees. In *ICML '22: Proceedings of the 39th International Conference on Machine Learning*, 2022.

Yossi Arjevani, Yair Carmon, John C. Duchi, Dylan J. Foster, Nathan Srebro, and Blake E. Woodworth. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, 199(1):165–214, 2023.

Amit Attia and Tomer Koren. SGD with AdaGrad stepsizes: Full adaptivity with high probability to unknown parameters, unbounded gradients and affine variance. In *International Conference on Machine Learning*, pp. 1147–1171. PMLR, 2023.

Amit Attia and Tomer Koren. How free is parameter-free stochastic optimization? In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24, 2024.

Atilim Gunes Baydin, Robert Cornish, David Martinez Rubio, Mark Schmidt, and Frank Wood. Online learning rate adaptation with hypergradient descent. In *International Conference on Learning Representations*, 2018.

Yair Carmon and Oliver Hinder. Making SGD parameter-free. In *Proceedings of Thirty Fifth Conference on Learning Theory*, pp. 2360–2389. PMLR, 2022.

Kartik Chandra, Audrey Xie, Jonathan Ragan-Kelley, and Erik Meijer. Gradient descent: The ultimate optimizer. *Advances in Neural Information Processing Systems*, 35:8214–8225, 2022.

Ya-Chi Chu, Wenzhi Gao, Yinyu Ye, and Madeleine Udell. Provable and practical online learning rate adaptation with hypergradient descent. In *Forty-second International Conference on Machine Learning*, 2025.

Ashok Cutkosky and Harsh Mehta. Momentum improves normalized sgd. In *International conference on machine learning*, pp. 2260–2268. PMLR, 2020.

Ashok Cutkosky and Francesco Orabona. Black-box reductions for parameter-free online learning in banach spaces. In *Conference On Learning Theory*, pp. 1493–1529. PMLR, 2018.

Ashok Cutkosky, Aaron Defazio, and Harsh Mehta. Mechanic: A learning rate tuner. *Advances in neural information processing systems*, 36:47828–47848, 2023a.

Ashok Cutkosky, Harsh Mehta, and Francesco Orabona. Optimal, stochastic, non-smooth, non-convex optimization through online-to-non-convex conversion. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023b.

Damek Davis, Dmitriy Drusvyatskiy, Sham Kakade, and Jason D. Lee. Stochastic subgradient method converges on tame functions. *Found. Comput. Math.*, 20(1):119–154, 2020.

Aaron Defazio and Konstantin Mishchenko. Learning-rate-free learning by D-adaptation. In *International Conference on Machine Learning*, pp. 7449–7479. PMLR, 2023.

Aaron Defazio, Ashok Cutkosky, Harsh Mehta, and Konstantin Mishchenko. Optimal linear decay learning rate schedules and further refinements. *arXiv preprint arXiv:2310.07831*, 2023.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Matthew Faw, Isidoros Tziotis, Constantine Caramanis, Aryan Mokhtari, Sanjay Shakkottai, and Rachel Ward. The power of adaptivity in SGD: Self-tuning step sizes with unbounded gradients and affine variance. In *Conference on Learning Theory*, pp. 313–355. PMLR, 2022.

Sébastien Gadat and Ioana Gavra. Asymptotic study of stochastic adaptive algorithms in non-convex landscape. *Journal of Machine Learning Research*, 23(228):1–54, 2022.

Wenzhi Gao, Ya-Chi Chu, Yinyu Ye, and Madeleine Udell. Gradient methods with online scaling. In *Proceedings of Thirty Eighth Conference on Learning Theory*, pp. 2192–2226. PMLR, 2025.

Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Maor Ivgi, Oliver Hinder, and Yair Carmon. DoG is SGD's best friend: A parameter-free dynamic step size schedule. In *International Conference on Machine Learning*, pp. 14465–14499. PMLR, 2023.

Yao Ji and Guanghui Lan. Stochastic auto-conditioned fast gradient methods with optimal rates. *arXiv preprint arXiv:2604.06525*, 2026.

Pooria Joulani, Anant Raj, Andras Gyorgy, and Csaba Szepesvari. A simpler approach to accelerated optimization: iterative averaging meets optimism. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 4984–4993. PMLR, 2020.

Ali Kavis, Kfir Y. Levy, Francis Bach, and Volkan Cevher. Unixgrad: A universal, adaptive algorithm with optimal guarantees for constrained optimization. In *Advances in Neural Information Processing Systems 32*, pp. 6260–6269, 2019.

Ali Kavis, Kfir Yehuda Levy, and Volkan Cevher. High probability bounds for a class of nonconvex algorithms with AdaGrad stepsize. In *International Conference on Learning Representations*, 2022.

Ahmed Khaled, Konstantin Mishchenko, and Chi Jin. DoWG unleashed: An efficient universal parameter-free gradient descent method. *Advances in Neural Information Processing Systems*, 36: 6748–6769, 2023.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference for Learning Representations (ICLR)*, 2015.

Itai Kreisler, Maor Ivgi, Oliver Hinder, and Yair Carmon. Accelerated parameter-free stochastic optimization. In *Proceedings of Thirty Seventh Conference on Learning Theory*, pp. 3257–3324. PMLR, 2024.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Kfir Y. Levy. Online to offline conversions, universality and adaptive minibatch sizes. In *Advances in Neural Information Processing Systems 30*, pp. 1613–1622, 2017.

Kfir Y Levy, Alp Yurtsever, and Volkan Cevher. Online adaptive methods, universality and acceleration. In *Neural and Information Processing Systems (NeurIPS)*, 2018.

Tianjiao Li and Guanghui Lan. A simple uniformly optimal method without line search for convex optimization. *Mathematical Programming*, pp. 1–38, 2025.

Xiaoyu Li and Francesco Orabona. On the convergence of stochastic gradient descent with adaptive stepsizes. In *The 22nd international conference on artificial intelligence and statistics*, pp. 983–992. PMLR, 2019.

Xiaoyu Li and Francesco Orabona. A high probability analysis of adaptive SGD with momentum. In *Workshop on Beyond First Order Methods in ML Systems at ICML'20*, 2020.

Zijian Liu, Ta Duy Nguyen, Alina Ene, and Huy Nguyen. On the convergence of AdaGrad(Norm) on $\mathbb{R}^d$: Beyond convexity, non-asymptotic rate and acceleration. In *The Eleventh International Conference on Learning Representations*, 2023a.

Zijian Liu, Ta Duy Nguyen, Thien Hang Nguyen, Alina Ene, and Huy Nguyen. High probability convergence of stochastic gradient methods. In *International Conference on Machine Learning*, pp. 21884–21914. PMLR, 2023b.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

Yura Malitsky and Konstantin Mishchenko. Adaptive gradient descent without descent. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20, 2020.

Yura Malitsky and Konstantin Mishchenko. Adaptive proximal gradient method for convex optimization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

H Brendan McMahan and Matthew Streeter. Adaptive bound optimization for online convex optimization. *COLT 2010*, pp. 244, 2010.

Konstantin Mishchenko and Aaron Defazio. Prodigy: An expeditiously adaptive parameter-free learner. In *International Conference on Machine Learning*, pp. 35779–35804. PMLR, 2024.

Mehryar Mohri and Scott Yang. Accelerating online convex optimization via adaptive prediction. In *Artificial Intelligence and Statistics*, pp. 848–856. PMLR, 2016.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729. IEEE, 2008.

Francesco Orabona. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

Francesco Orabona and Dávid Pál. Coin betting and parameter-free online learning. *Advances in Neural Information Processing Systems*, 29, 2016.

Francesco Orabona and Tatiana Tommasi. Training deep networks without learning rates through coin betting. *Advances in neural information processing systems*, 30, 2017.

Kaan Ozkara, Can Karakus, Parameswaran Raman, Mingyi Hong, Shoham Sabach, Branislav Kveton, and Volkan Cevher. Mada: Meta-adaptive optimizers through hyper-gradient descent. In *Forty-first International Conference on Machine Learning*, 2024.

Alexander Rakhlin and Karthik Sridharan. Online learning with predictable sequences. In *Conference on Learning Theory*, pp. 993–1019. PMLR, 2013.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

Anton Rodomanov, Ali Kavis, Yongtao Wu, Kimon Antonakopoulos, and Volkan Cevher. Universal gradient methods for stochastic convex optimization. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24, 2024.

David Martınez Rubio. Convergence analysis of an adaptive method of gradient descent. *University of Oxford, Oxford, M. Sc. thesis*, 2017.

Stanford CS231n. Tiny imagenet visual recognition challenge. `https://tiny-imagenet.herokuapp.com/`, 2015. Stanford CS231n Course Project.

Jacob Steinhardt and Percy Liang. Adaptivity and optimism: An improved exponentiated gradient algorithm. In *International conference on machine learning*, pp. 1593–1601. PMLR, 2014.

Tijmen Tieleman and G Hinton. Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning. *Technical Report*, 2012.

Rachel Ward, Xiaoxia Wu, and Leon Bottou. Adagrad stepsizes: Sharp convergence over nonconvex landscapes. *Journal of Machine Learning Research*, 21(219):1–30, 2020.

Kaiyue Wen, David Hall, Tengyu Ma, and Percy Liang. Fantastic pretraining optimizers and where to find them. *arXiv preprint arXiv:2509.02046*, 2025.

Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Jingzhao Zhang, Hongzhou Lin, Stefanie Jegelka, Suvrit Sra, and Ali Jadbabaie. Complexity of finding stationary points of nonconvex nonsmooth functions. In *International Conference on Machine Learning*, pp. 11173–11182. PMLR, 2020.

Zhenxun Zhuang, Ashok Cutkosky, and Francesco Orabona. Surrogate losses for online learning of stepsizes in stochastic non-convex optimization. In *International Conference on Machine Learning*, pp. 7664–7672. PMLR, 2019.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, pp. 928–935, 2003.

APPENDIX

A EXTENDED RELATED WORK

Classical stochastic first-order methods. Dating back to the seminal work Robbins & Monro (1951), the theoretical behavior of SGD and its many variants have been extensively studied. Considering general smooth functions, it is well-known that the learning rate must decrease at a rate of $\eta_t = O(1/\sqrt{t})$ where t is the iteration counter, and must also satisfy $\eta_t \leq O(1/L)$. Ghadimi & Lan (2013) established that SGD with a properly chosen learning rate achieves a complexity of $\mathcal{O}(\epsilon^{-2} + \sigma^2\epsilon^{-4})$, interpolating between deterministic and stochastic rates and matching the lower bounds (Arjevani et al., 2023). However, the choice of the learning rate depends on the problem parameters, e.g., L and σ, which are typically unknown or difficult to estimate in practice. Similar requirements are in place for the learning rate when the loss is ρ-weakly convex (Davis et al., 2020).

Adaptive and parameter-free optimization methods. AdaGrad was introduced in two concurrent works McMahan & Streeter (2010); Duchi et al. (2011) for minimizing a sequence of *online* convex losses. The main idea is to compute a time-varying learning rate by accumulating squared norms of stochastic gradients. This fundamental idea paved the way for many algorithms such as Adam (Kingma & Ba, 2015), RMSProp Tieleman & Hinton (2012), Adadelta Zeiler (2012), and their variants, which demonstrate strong empirical performance. Beyond the online optimization setup, they have been shown to automatically adapt to problem-dependent parameters such as smoothness, noise variance, and bounds on gradients. Their convergence properties have been well-studied for the convex setting (Levy, 2017; Levy et al., 2018; Kavis et al., 2019; Joulani et al., 2020; Antonakopoulos et al., 2022; Liu et al., 2023a; Rodomanov et al., 2024) and non-convex setting (Li & Orabona, 2019; Ward et al., 2020; Li & Orabona, 2020; Kavis et al., 2022; Gadat & Gavra, 2022; Faw et al., 2022; Attia & Koren, 2023; Liu et al., 2023b).

A downside of the above-mentioned methods is sensitivity to the initial learning rate due to the need to know an estimate of the initial distance, $\|\boldsymbol{x}_0 - \boldsymbol{x}^*\|$. Recently, *parameter-free* optimization (Carmon & Hinder, 2022; Ivgi et al., 2023; Defazio & Mishchenko, 2023; Khaled et al., 2023; Mishchenko & Defazio, 2024; Kreisler et al., 2024; Attia & Koren, 2024) has gained popularity with a focus on augmenting robustness. Essentially, they combine an AdaGrad-type learning rate with a certificate that estimates the initial distance to the solution. Although this helps the learning rate increase from the initial value, the scaling factor is practically bounded, restricting flexibility. On a related front, a different line of work Malitsky & Mishchenko (2020; 2024); Li & Lan (2025) studied parameter-free gradient methods with local curvature estimation for convex, deterministic problems. They differ from AdaGrad-type methods in having a non-monotone learning rate that estimates the time evolution of local smoothness. A downside of these methods is empirical instability; if the increasing behavior is not tamed properly, optimization performance could be unstable.

Hypergradient descent. Originally proposed as a heuristic for stochastic optimization in Almeida et al. (1999), hypergradient descent updates the learning rate by computing the gradient with respect to the learning rate. It was later rediscovered and adapted to modern deep learning Rubio (2017); Baydin et al. (2018), with several subsequent works refining this approach Chandra et al. (2022); Ozkara et al. (2024). Recently, Gao et al. (2025); Chu et al. (2025) provided convergence guarantees from an online learning perspective, but their analysis is limited to deterministic convex settings.

Online learning-guided methods. Focusing on parameter-free online learning Orabona & Pál (2016), Orabona & Tommasi (2017) reformulated SGD as a coin-betting game and applied a betting algorithm to eliminate the need to manually tune the learning rate. They provided convergence guarantees for convex and quasi-convex objectives. Cutkosky et al. (2023a) proposed a technique for adaptively scaling any base optimization algorithm and learning rate schedule, which is grounded in a black-box reduction framework Cutkosky & Orabona (2018). The work most relevant to ours is that of Zhuang et al. (2019), who consider non-convex stochastic optimization and introduce a surrogate loss technique for learning rate update. However, their method requires knowledge of problem-dependent parameters (e.g., gradient's Lipschitz constant), thereby limiting flexibility.

B Proof of Lemma 3.1

Recall from Eq. 4 that $F(\boldsymbol{x}_{t+1}) - F(\boldsymbol{x}_t) = -\eta_t \mathbb{E}_{\lambda_t, \xi_t'}[\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle]$. We now decompose the right-hand side as $\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle = \langle \boldsymbol{g}_t'(\boldsymbol{x}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle + \langle \boldsymbol{g}_t'(\boldsymbol{w}_t) - \boldsymbol{g}_t'(\boldsymbol{x}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle$, where $\boldsymbol{g}_t'(\boldsymbol{x}_t) = \nabla f(\boldsymbol{x}_t; \xi_t')$. For the first term, since ξ_t and ξ_t' are independent samples from the distribution $\mathcal{D}$, it holds that $\mathbb{E}[\langle \boldsymbol{g}_t'(\boldsymbol{x}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle] = \mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|^2]$. Moreover, for the second term, it follows from Cauchy-Schwarz inequality and the definition of L_t that

$$\langle \boldsymbol{g}_t'(\boldsymbol{w}_t) - \boldsymbol{g}_t'(\boldsymbol{x}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle \geq -\|\boldsymbol{g}_t'(\boldsymbol{w}_t) - \boldsymbol{g}_t'(\boldsymbol{x}_t)\|\|\boldsymbol{g}_t(\boldsymbol{x}_t)\| = -L_t\|\boldsymbol{w}_t - \boldsymbol{x}_t\|\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|.$$

Since $\boldsymbol{w}_t = \boldsymbol{x}_t + \lambda_t(\boldsymbol{x}_{t+1} - \boldsymbol{x}_t)$ and $\lambda_t \in [0, 1]$, we further have $\|\boldsymbol{w}_t - \boldsymbol{x}_t\| = \lambda_t\|\boldsymbol{x}_{t+1} - \boldsymbol{x}_t\| \leq \|\boldsymbol{x}_{t+1} - \boldsymbol{x}_t\| = \eta_t\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|$, which leads to $\langle \boldsymbol{g}_t'(\boldsymbol{w}_t) - \boldsymbol{g}_t'(\boldsymbol{x}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle \geq -L_t\eta_t\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2$. By combining both results, we obtain that

$$\mathbb{E}[\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle] \geq \mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|^2] - \mathbb{E}[L_t\eta_t\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2].$$

Hence, taking expectations on both sides of Eq. 4, we can write

$$\begin{aligned}
\mathbb{E}[F(\boldsymbol{x}_{t+1}) - F(\boldsymbol{x}_t)] &= -\mathbb{E}[\eta_t\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle] \\
&= -\mathbb{E}[(\eta_t - \eta)\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle] - \eta\mathbb{E}[\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle] \\
&\leq -\mathbb{E}[(\eta_t - \eta)\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle] - \eta\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|^2] + \eta\mathbb{E}[L_t\eta_t\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2].
\end{aligned} \tag{11}$$

Moreover, from Young's inequality $\eta\eta_t \leq \frac{\eta^2}{2} + \frac{\eta_t^2}{2}$, the last term in Eq. 11 can be bounded by $L_t\eta\eta_t\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2 \leq \frac{L_t\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2\eta^2}{2} + \frac{L_t\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2\eta_t^2}{2}$. Thus, we obtain

$$\begin{aligned}
\mathbb{E}[F(\boldsymbol{x}_{t+1}) - F(\boldsymbol{x}_t)] &\leq -\eta\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|^2] - \mathbb{E}[(\eta_t - \eta)\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \boldsymbol{g}_t(\boldsymbol{x}_t)\rangle] + \mathbb{E}[L_t\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2\eta^2] \\
&\quad + \mathbb{E}\left[\frac{L_t\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2\eta_t^2}{2} - \frac{L_t\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2\eta^2}{2}\right] \\
&= -\eta\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|^2] + \eta^2\mathbb{E}[L_t\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2] + \mathbb{E}[\ell_t(\eta_t) - \ell_t(\eta)],
\end{aligned} \tag{12}$$

where in the last equality we used the definition of the surrogate loss function in Eq. 5. Moreover, Since $L_t \leq L^{\max}$ for any $t \geq 0$ with probability one, we have $\mathbb{E}[L_t\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2] \leq L^{\max}\mathbb{E}[\|\boldsymbol{g}_t(\boldsymbol{x}_t)\|^2] \leq L^{\max}(\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|^2] + \sigma^2)$. Plugging this bound in Eq. 12 and rearranging, we obtain

$$\mathbb{E}[(\eta - \eta^2 L^{\max})\|\nabla F(\boldsymbol{x}_t)\|^2] \leq \mathbb{E}[F(\boldsymbol{x}_t) - F(\boldsymbol{x}_{t+1})] + L^{\max}\eta^2\sigma^2 + \mathbb{E}[\ell_t(\eta_t) - \ell_t(\eta)].$$

Summing the above inequality from $t = 0$ to $t = T - 1$ yields the desired result. This completes the proof.

C Proof of Theorem 4.2

We divide the proof of Theorem 4.2 into the following three steps.

Step 1: Following similar arguments as in the proof of Lemma 3.1, we first bound the function value decrease after one iteration. Its proof can be found in Section C.1.

Lemma C.1. *For any $\eta > 0$, we have* $\mathbb{E}[F(\boldsymbol{x}_{t+1}) - F(\boldsymbol{x}_t)] \leq \mathbb{E}\Big[-\frac{\eta}{3}\|\nabla F(\boldsymbol{x}_t)\| + \max\{L_t, M\}\eta^2 + \frac{8\eta}{3}\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\|\Big] + \mathbb{E}\Big[-(\eta_t - \eta)\Big\langle \boldsymbol{g}'(\boldsymbol{w}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\Big\rangle + \frac{\max\{L_t, M\}}{2}(\eta_t^2 - \eta^2)\Big]$.

In the above bound, the first bracketed term shows up in the analysis of normalized SGD with momentum in Cutkosky & Mehta (2020); it is the upper bound we get when choosing $\eta_t = \eta$. Moreover, the second term in the bracket captures the difference between the actual learning rate η_t and the comparator η. It will be incorporated into the surrogate loss function and be bounded by the regret.

Step 2: Next, we control the approximation error $\mathbb{E}[\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\|]$ incurred by exponential moving averaging.

Lemma C.2. *Define* $\tilde{L}_t = \frac{\|\boldsymbol{g}'_t(\boldsymbol{x}_{t+1}) - \boldsymbol{g}'_t(\boldsymbol{x}_t)\|}{\|\boldsymbol{x}_{t+1} - \boldsymbol{x}_t\|}$. *Then we have* $\sum_{t=0}^{T-1} \mathbb{E}[\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\|] \leq \frac{\sigma}{\alpha} + \sigma\sqrt{\alpha}T + \frac{1-\alpha}{\alpha}\sum_{t=0}^{T-2}\mathbb{E}[\tilde{L}_t\eta_t]$.

Lemma C.2 upper bounds the approximation error in terms of the learning rate η_t. As we shall see in the next step, this term will also be incorporated into our surrogate loss and be bounded by the regret.

Step 3: By summing the inequality in Lemma C.1 from $t = 0$ to $t = T - 1$ and applying Lemma C.2, we obtain

$$
\begin{aligned}
&\mathbb{E}[F(\boldsymbol{x}_T) - F(\boldsymbol{x}_0)] \\
&\leq -\frac{\eta}{3}\mathbb{E}\Big[\sum_{t=0}^{T-1}\|\nabla F(\boldsymbol{x}_t)\|\Big] + \mathbb{E}\Big[\sum_{t=0}^{T-1}\max\{L_t, M\}\Big]\eta^2 + \frac{8\eta}{3}\mathbb{E}\Big[\sum_{t=0}^{T-1}\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\|\Big] \\
&\quad + \sum_{t=0}^{T-1}\mathbb{E}\Big[-(\eta_t - \eta)\Big\langle \boldsymbol{g}'(\boldsymbol{w}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\Big\rangle + \frac{\max\{L_t, M\}}{2}(\eta_t^2 - \eta^2)\Big] \\
&\leq -\frac{\eta}{3}\mathbb{E}\Big[\sum_{t=0}^{T-1}\|\nabla F(\boldsymbol{x}_t)\|\Big] + \mathbb{E}\Big[\sum_{t=0}^{T-1}\max\{L_t, M\}\Big]\eta^2 + \frac{8\eta}{3}(\frac{\sigma}{\alpha} + \sigma\sqrt{\alpha}T) + \frac{8(1-\alpha)}{3\alpha}\mathbb{E}\Big[\sum_{t=0}^{T-2}\tilde{L}_t\eta_t\eta\Big] \\
&\quad + \sum_{t=0}^{T-1}\mathbb{E}\Big[-(\eta_t - \eta)\Big\langle \boldsymbol{g}'(\boldsymbol{w}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\Big\rangle + \frac{\max\{L_t, M\}}{2}(\eta_t^2 - \eta^2)\Big].
\end{aligned}
$$

Moreover, by Young's inequality, we have $\tilde{L}_t\eta_t\eta \leq \frac{\tilde{L}_t}{2}\eta_t^2 + \frac{\tilde{L}_t}{2}\eta^2 = \tilde{L}_t\eta^2 + (\frac{\tilde{L}_t}{2}\eta_t^2 - \frac{\tilde{L}_t}{2}\eta^2)$. Using $\Delta_F = F(\boldsymbol{x}_0) - F(\boldsymbol{x}^*) \geq F(\boldsymbol{x}_0) - F(\boldsymbol{x}_T)$ and recalling the definition of ℓ_t^{N} in Eq. 8, we obtain

$$
\begin{aligned}
0 \leq -\frac{\eta}{3}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|] + \frac{8\eta}{3}(\frac{\sigma}{\alpha} + \sigma\sqrt{\alpha}T) + \mathbb{E}\Big[\sum_{t=0}^{T-1}\max\{L_t, M\} + \frac{8(1-\alpha)\sum_{t=0}^{T-1}\tilde{L}_t}{3\alpha}\Big]\eta^2 \\
+ \mathbb{E}[\sum_{t=0}^{T-1}(\ell_t^{\mathrm{N}}(\eta_t) - \ell_t^{\mathrm{N}}(\eta))] + \mathbb{E}[\Delta_F].
\end{aligned}
$$

Now for any $\eta \in [0, \eta^{\max}]$, we can upper bound $\sum_{t=0}^{T-1}(\ell_t^{\mathrm{N}}(\eta_t) - \ell_t^{\mathrm{N}}(\eta)) \leq \mathrm{Reg}_T^{\mathrm{N}}$ by definition, and hence we can choose the value of η freely from the interval $[0, \eta^{\max}]$ in the above bound. We now consider the following cases:

(i) **Case I:** we have $\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|] \leq 16(\frac{\sigma}{\alpha} + \sigma\sqrt{\alpha}T)$;

(ii) **Case II:** we have $\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|] \geq 16(\frac{\sigma}{\alpha} + \sigma\sqrt{\alpha}T)$. This further implies that

$$
0 \leq -\frac{\eta}{6}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|] + \mathbb{E}\Big[\sum_{t=0}^{T-1}\max\{L_t, M\} + \frac{8(1-\alpha)\sum_{t=0}^{T-1}\tilde{L}_t}{3\alpha}\Big]\eta^2 + \mathbb{E}[\mathrm{Reg}_T^{\mathrm{N}} + \Delta_F]. \tag{13}
$$

Moreover, we set the value of η as

$$
\eta = \min\left\{\frac{\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|]}{\mathbb{E}[12\sum_{t=0}^{T-1}\max\{L_t, M\} + \frac{32(1-\alpha)}{\alpha}\sum_{t=0}^{T-1}\tilde{L}_t]}, \eta^{\max}\right\}. \tag{14}
$$

This again leads to two subcases depending on the value of η:

- If η takes the first value in Eq. 14, we obtain from Eq. 13 that

$$
\frac{1}{12}\frac{(\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|])^2}{\mathbb{E}[12\sum_{t=0}^{T-1}\max\{L_t, M\} + \frac{32(1-\alpha)}{\alpha}\sum_{t=0}^{T-1}\tilde{L}_t]} \leq \mathbb{E}[\mathrm{Reg}_T^{\mathrm{N}} + \Delta_F].
$$

To simplify the notation, let $\mathcal{B} = \Delta_F + \mathrm{Reg}_T^{\mathrm{N}}$. With some algebraic manipulation and using the fact that $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$, we obtain

$$
\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|] \leq 12\sqrt{\mathbb{E}[\mathcal{B}]\mathbb{E}\Big[\sum_{t=0}^{T-1}\max\{L_t, M\}\Big]} + 8\sqrt{\frac{6(1-\alpha)}{\alpha}}\sqrt{\mathbb{E}[\mathcal{B}]\mathbb{E}\Big[\sum_{t=0}^{T-1}\tilde{L}_t\Big]}.
$$

- If η takes the second value in Eq. 14, then $\eta = \eta^{\max} \leq \frac{\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|]}{\mathbb{E}[12\sum_{t=0}^{T-1}\max\{L_t,M\}+\frac{32(1-\alpha)}{\alpha}\sum_{t=0}^{T-1}\tilde{L}_t]}$. In this case, we obtain from Eq. 13 that

$$\frac{\eta^{\max}}{12}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|] \leq \mathbb{E}[\mathcal{B}] \quad \Rightarrow \quad \sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|] \leq \frac{12\mathbb{E}[\mathcal{B}]}{\eta^{\max}}.$$

Combining the upper bounds in all cases and using the definition of L_T^{avg}, we can deduce that

$$\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|] \leq 16(\frac{\sigma}{\alpha}+\sigma\sqrt{\alpha}T)+12\sqrt{\mathbb{E}[\mathcal{B}]L_T^{\text{avg}}T}+8\sqrt{\frac{6(1-\alpha)}{\alpha}}\sqrt{\mathbb{E}[\mathcal{B}]L_T^{\text{avg}}T}+\frac{12\mathbb{E}[\mathcal{B}]}{\eta^{\max}}. \tag{15}$$

Finally, we can choose the parameter α to optimize the above upper bound. Specifically, we let

$$\alpha = \min\Big\{\frac{1}{\sigma\sqrt{T}}, 1\Big\}.$$

If $\frac{1}{\sigma\sqrt{T}} \leq 1$, then we have $\frac{16\sigma}{\alpha} \leq 16\sigma^2\sqrt{T}$, $16\sigma\sqrt{\alpha}T \leq 16\sigma^{1/2}T^{3/4}$, and $8\sqrt{\frac{6(1-\alpha)}{\alpha}}\sqrt{\mathbb{E}[\mathcal{B}]L_T^{\text{avg}}T} \leq 8\sqrt{6}\sigma^{1/2}(L_T^{\text{avg}}\mathbb{E}[\mathcal{B}])^{1/2}T^{3/4}$. Otherwise, if $\frac{1}{\sigma\sqrt{T}} > 1$, then $\alpha = 1$ and we have $\frac{16\sigma}{\alpha} = 16\sigma \leq \frac{16}{\sqrt{T}}$, $16\sigma\sqrt{\alpha}T \leq 16\sigma T \leq 16\sqrt{T}$, and $8\sqrt{\frac{6(1-\alpha)}{\alpha}}\sqrt{\mathbb{E}[\mathcal{B}]L_T^{\text{avg}}T} = 0$. Hence, combining both cases, we conclude that

$$\begin{aligned}&16(\frac{\sigma}{\alpha}+\sigma\sqrt{\alpha}T)+8\sqrt{\frac{6(1-\alpha)}{\alpha}}\sqrt{\mathbb{E}[\mathcal{B}]L_T^{\text{avg}}T}\\ &\leq 16(\sigma^2+2)\sqrt{T}+(16+8\sqrt{6}(L_T^{\text{avg}}\mathbb{E}[\mathcal{B}])^{1/2})\sigma^{1/2}T^{3/4}.\end{aligned}$$

By using the above bound and dividing both sides by T in Eq. 15, we arrive at

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(\boldsymbol{x}_t)\|] \leq{}& \frac{(16+8\sqrt{6}(L_T^{\text{avg}}\mathbb{E}[\mathcal{B}])^{1/2})\sigma^{1/2}}{T^{1/4}}+\frac{16(\sigma^2+2)}{\sqrt{T}}+12\frac{\sqrt{L_T^{\text{avg}}\mathbb{E}[\mathcal{B}]}}{\sqrt{T}}\\ &+\frac{12\mathbb{E}[\mathcal{B}]}{\eta^{\max}T}.\end{aligned}$$

This completes the proof of Theorem 4.2.

C.1 Proof of Lemma C.1

Similar to the arguments in Section 3, we first apply the fundamental theorem of calculus to get $F(\boldsymbol{x}_{t+1}) - F(\boldsymbol{x}_t) = \langle \boldsymbol{\nabla}_t, \boldsymbol{x}_{t+1} - \boldsymbol{x}_t\rangle = -\eta_t\langle \boldsymbol{\nabla}_t, \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\rangle$. Since $\boldsymbol{\nabla}_t = \mathbb{E}_{\lambda_t}[\nabla F(\boldsymbol{w}_t)] = \mathbb{E}_{\lambda_t,\xi_t'}[\boldsymbol{g}_t'(\boldsymbol{w}_t)]$, we further have

$$F(\boldsymbol{x}_{t+1}) - F(\boldsymbol{x}_t) = -\eta_t\mathbb{E}_{\lambda_t,\xi_t'}\Big[\Big\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\Big\rangle\Big]. \tag{16}$$

Next, we decompose the right-hand side of Eq. 16 as

$$\begin{aligned}\Big\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\Big\rangle &= \Big\langle \boldsymbol{g}_t'(\boldsymbol{x}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\Big\rangle + \Big\langle \boldsymbol{g}_t'(\boldsymbol{w}_t) - \boldsymbol{g}_t'(\boldsymbol{x}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\Big\rangle\\ &\geq \Big\langle \boldsymbol{g}_t'(\boldsymbol{x}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\Big\rangle - \|\boldsymbol{g}_t'(\boldsymbol{w}_t) - \boldsymbol{g}_t'(\boldsymbol{x}_t)\|,\end{aligned}$$

where we used Cauchy-Schwarz inequality in the last step. Using the definition of L_t, we have

$$\|\boldsymbol{g}_t'(\boldsymbol{w}_t) - \boldsymbol{g}_t'(\boldsymbol{x}_t)\| \leq L_t\|\boldsymbol{w}_t - \boldsymbol{x}_t\| = L_t\lambda_t\|\boldsymbol{x}_{t+1} - \boldsymbol{x}_t\| \leq L_t\eta_t \leq \max\{L_t, M\}\eta_t. \tag{17}$$

Moreover, since $\boldsymbol{g}_t'(\boldsymbol{x}_t)$ and $\boldsymbol{m}_t$ are independent conditioned on $\boldsymbol{x}_t$, we further have $\mathbb{E}[\langle \boldsymbol{g}_t'(\boldsymbol{x}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\rangle] = \mathbb{E}[\langle \nabla F(\boldsymbol{x}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\rangle]$, which is further lower bounded in the following lemma.

Lemma C.3. *We have* $\langle \nabla F(\boldsymbol{x}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\rangle \geq \frac{1}{3}\|\nabla F(\boldsymbol{x}_t)\| - \frac{8}{3}\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\|$.

Proof. Our proof is inspired by (Cutkosky & Mehta, 2020, Lemma 2). We consider two cases:

(i) If $\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\| \leq \frac{1}{2}\|\nabla F(\boldsymbol{x}_t)\|$, then by the triangle inequality, we have $\|\boldsymbol{m}_t\| \leq \frac{3}{2}\|\nabla F(\boldsymbol{x}_t)\|$. Therefore, we have

$$\begin{aligned}\frac{1}{\|\boldsymbol{m}_t\|}\langle \nabla F(\boldsymbol{x}_t), \boldsymbol{m}_t\rangle &= \frac{1}{\|\boldsymbol{m}_t\|}(\|\nabla F(\boldsymbol{x}_t)\|^2 + \langle \nabla F(\boldsymbol{x}_t), \boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\rangle) \\ &\geq \frac{1}{\|\boldsymbol{m}_t\|}(\|\nabla F(\boldsymbol{x}_t)\|^2 - \frac{1}{2}\|\nabla F(\boldsymbol{x}_t)\|^2) \geq \frac{1}{3}\|\nabla F(\boldsymbol{x}_t)\|,\end{aligned}$$

where we used $\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\| \leq \frac{1}{2}\|\nabla F(\boldsymbol{x}_t)\|$ in the first inequality and $\|\boldsymbol{m}_t\| \leq \frac{3}{2}\|\nabla F(\boldsymbol{x}_t)\|$ in the second one. Since $\frac{8}{3}\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\| \geq 0$, the result in Lemma C.3 holds under this case.

(ii) Otherwise, if $\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\| > \frac{1}{2}\|\nabla F(\boldsymbol{x}_t)\|$, we can instead use Cauchy-Schwarz inequality to bound

$$\begin{aligned}\frac{1}{\|\boldsymbol{m}_t\|}\langle \nabla F(\boldsymbol{x}_t), \boldsymbol{m}_t\rangle \geq -\|\nabla F(\boldsymbol{x}_t)\| &= \frac{1}{3}\|\nabla F(\boldsymbol{x}_t)\| - \frac{4}{3}\|\nabla F(\boldsymbol{x}_t)\| \\ &\geq \frac{1}{3}\|\nabla F(\boldsymbol{x}_t)\| - \frac{8}{3}\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\|,\end{aligned}$$

where we used $\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\| > \frac{1}{2}\|\nabla F(\boldsymbol{x}_t)\|$ in the last inequality.

This completes the proof. □

Combining Eq. 17 and Lemma C.3, we obtain that

$$\mathbb{E}\Big[\Big\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\Big\rangle\Big] \geq \mathbb{E}\Big[\frac{1}{3}\|\nabla F(\boldsymbol{x}_t)\| - \frac{8}{3}\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\| - \max\{L_t, M\}\eta_t\Big].$$

Hence, it further follows from Eq. 16 that

$$\begin{aligned}\mathbb{E}[F(\boldsymbol{x}_{t+1}) - F(\boldsymbol{x}_t)] &= -\mathbb{E}\Big[(\eta_t - \eta)\Big\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\Big\rangle\Big] - \eta\mathbb{E}\Big[\Big\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\Big\rangle\Big] \\ &\leq -\mathbb{E}\Big[(\eta_t - \eta)\Big\langle \boldsymbol{g}_t'(\boldsymbol{w}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\Big\rangle - \frac{\eta\|\nabla F(\boldsymbol{x}_t)\|}{3} + \frac{8\eta\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\|}{3} \\ &\quad + \max\{L_t, M\}\eta_t\eta\Big].\end{aligned}$$

By using Young's inequality $\eta_t\eta \leq \frac{\eta_t^2}{2} + \frac{\eta^2}{2}$ and rearranging, we obtain the inequality in Lemma C.1.

C.2 Proof of Lemma C.2

From the update rule in Eq. 7, we can write

$$\begin{aligned}\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t) = (1-\alpha)(\boldsymbol{m}_{t-1} - \nabla F(\boldsymbol{x}_{t-1})) + \alpha(\nabla f(\boldsymbol{x}_t; \xi_t) - \nabla F(\boldsymbol{x}_t)) \\ + (1-\alpha)(\nabla F(\boldsymbol{x}_{t-1}) - \nabla F(\boldsymbol{x}_t)).\end{aligned} \tag{18}$$

Define the stochastic gradient error $\boldsymbol{e}_t = \nabla f(\boldsymbol{x}_t; \xi_t) - \nabla F(\boldsymbol{x}_t)$. By Assumption 2.1, we have $\mathbb{E}[\boldsymbol{e}_t] = 0$ and $\mathbb{E}[\|\boldsymbol{e}_t\|^2] \leq \sigma^2$. Moreover, by multiplying both sides of Eq. 18 with $(1-\alpha)^{-t}$, we have

$$\begin{aligned}(\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t))(1-\alpha)^{-t} = (\boldsymbol{m}_{t-1} - \nabla F(\boldsymbol{x}_{t-1}))(1-\alpha)^{-t+1} + \alpha\boldsymbol{e}_t(1-\alpha)^{-t} \\ + (\nabla F(\boldsymbol{x}_{t-1}) - \nabla F(\boldsymbol{x}_t))(1-\alpha)^{-t+1}.\end{aligned}$$

Note that we set $\boldsymbol{m}_0 = \nabla f(\boldsymbol{x}_0; \xi_0)$. Thus, by summing the above inequality, we obtain

$$(\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t))(1-\alpha)^{-t} = \boldsymbol{e}_0 + \sum_{s=1}^{t}\boldsymbol{e}_s\alpha(1-\alpha)^{-s} + \sum_{s=1}^{t}(\nabla F(\boldsymbol{x}_{s-1}) - \nabla F(\boldsymbol{x}_s))(1-\alpha)^{-s+1}.$$

Therefore, it follows from the triangle inequality that

$$\|\boldsymbol{m}_t-\nabla F(\boldsymbol{x}_t)\| \le \|\boldsymbol{e}_0\|(1-\alpha)^t+\left\|\sum_{s=1}^{t}\boldsymbol{e}_s\alpha(1-\alpha)^{t-s}\right\|+\sum_{s=1}^{t}\|\nabla F(\boldsymbol{x}_{s-1})-\nabla F(\boldsymbol{x}_s)\|(1-\alpha)^{t-s+1}. \tag{19}$$

By Jensen's inequality and the fact that $\{\xi_s\}_{s=1}^t$ are i.i.d. sampled from $\mathcal{D}$, we have $\mathbb{E}[\|\boldsymbol{e}_0\|] \le \sqrt{\mathbb{E}[\|\boldsymbol{e}_0\|^2]} = \sigma$ and

$$\mathbb{E}\left\|\sum_{s=1}^{t}\boldsymbol{e}_s\alpha(1-\alpha)^{t-s}\right\| \le \sqrt{\mathbb{E}\left\|\sum_{s=1}^{t}\boldsymbol{e}_s\alpha(1-\alpha)^{t-s}\right\|^2} \le \sqrt{\sum_{s=1}^{t}\sigma^2\alpha^2(1-\alpha)^{2(t-s)}}.$$

Moreover, it also follows from Jensen's inequality that $\mathbb{E}[\|\nabla F(\boldsymbol{x}_{s-1}) - \nabla F(\boldsymbol{x}_s)\|] \le \mathbb{E}[\|\nabla f(\boldsymbol{x}_s;\xi'_{s-1}) - \nabla f(\boldsymbol{x}_{s-1};\xi'_{s-1})\|] = \mathbb{E}[\tilde{L}_{s-1}\|\boldsymbol{x}_s - \boldsymbol{x}_{s-1}\|] = \mathbb{E}[\tilde{L}_{s-1}\eta_{s-1}]$. Hence, by taking the expectation on both sides of Eq. 19, we further have

$$\begin{aligned}
\mathbb{E}[\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\|] &\le \sigma(1-\alpha)^t + \sigma\alpha\sqrt{\sum_{s=1}^{t}(1-\alpha)^{2(t-s)}} + \sum_{s=1}^{t}\mathbb{E}[\tilde{L}_{s-1}\eta_{s-1}](1-\alpha)^{t-s+1} \\
&\le \sigma(1-\alpha)^t + \sigma\alpha\sqrt{\frac{1}{1-(1-\alpha)^2}} + \sum_{s=0}^{t-1}\mathbb{E}[\tilde{L}_s\eta_s](1-\alpha)^{t-s} \\
&\le \sigma(1-\alpha)^t + \sigma\sqrt{\alpha} + \sum_{s=0}^{t-1}\mathbb{E}[\tilde{L}_s\eta_s](1-\alpha)^{t-s}.
\end{aligned}$$

By summing the above inequality from $t=0$ to $t=T-1$, we obtain that

$$\sum_{t=0}^{T-1}\mathbb{E}[\|\boldsymbol{m}_t - \nabla F(\boldsymbol{x}_t)\|] \le \sigma\sum_{t=0}^{T-1}(1-\alpha)^t + \sigma\sqrt{\alpha}T + \sum_{t=0}^{T-1}\sum_{s=0}^{t-1}\mathbb{E}[\tilde{L}_s\eta_s](1-\alpha)^{t-s}$$

Since $\sum_{t=0}^{T-1}(1-\alpha)^t \le \frac{1}{\alpha}$ and $\sum_{t=0}^{T-1}\sum_{s=0}^{t-1}\mathbb{E}[\tilde{L}_s\eta_s](1-\alpha)^{t-s} = \sum_{s=0}^{T-2}\sum_{t=s+1}^{T-1}\mathbb{E}[\tilde{L}_s\eta_s](1-\alpha)^{t-s} \le \frac{1-\alpha}{\alpha}\sum_{s=0}^{T-2}\mathbb{E}[\tilde{L}_s\eta_s]$, we obtain Lemma C.2.

D Proof of Lemma 4.3

As discussed in Section 4, our update for η can be viewed as an instance of the optimistic FTRL algorithm. Therefore, we can invoke the convergence bound in (Orabona, 2019, Theorem 7.39), where $\psi_1 = \cdots = \psi_T = \frac{\delta}{2}\eta^2$ and $\tilde{\ell}_{t+1}(\eta) = -\eta\langle \boldsymbol{g}_{t+1}(\boldsymbol{x}_{t+1}), \frac{\boldsymbol{m}_{t+1}}{\|\boldsymbol{m}_{t+1}\|}\rangle$. Moreover, note that $\frac{\delta}{2}\eta^2 + \sum_{s=0}^{t}\ell_t^{\mathrm{N}}(\eta)$ is $(\delta + \sum_{s=0}^{t}(\max\{L_s, M\} + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_s))$-strongly convex, and $|(\ell_t^{\mathrm{N}})'(\eta_t) - \tilde{\ell}'_t(\eta_t)| = |-\langle \boldsymbol{g}'_t(\boldsymbol{w}_t) - \boldsymbol{g}_t(\boldsymbol{x}_t), \frac{\boldsymbol{m}_t}{\|\boldsymbol{m}_t\|}\rangle + \max\{L_t, M\}\eta_t + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_t\eta_t| \le \|\boldsymbol{g}'_t(\boldsymbol{w}_t) - \boldsymbol{g}_t(\boldsymbol{x}_t)\| + \max\{L_t, M\}\eta_t + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_t\eta_t$. Hence, we have

$$\sum_{t=0}^{T-1}(\ell_t(\eta_t) - \ell_t(\eta)) \le \frac{\delta}{2}\eta^2 + \sum_{t=0}^{T-1}\frac{(\max\{L_t, M\}\eta_t + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_t\eta_t + \|\boldsymbol{g}'_t(\boldsymbol{w}_t) - \boldsymbol{g}_t(\boldsymbol{x}_t)\|)^2}{2\delta + 2\sum_{s=0}^{t}(\max\{L_s, M\} + \frac{8(1-\alpha)\tilde{L}_s}{3\alpha})}. \tag{20}$$

Moreover, by the triangle inequality and the definition of L_t, we have $\|\boldsymbol{g}'_t(\boldsymbol{w}_t) - \boldsymbol{g}_t(\boldsymbol{x}_t)\| = \|\nabla f(\boldsymbol{w}_t;\xi'_t) - \nabla f(\boldsymbol{x}_t;\xi_t)\| \le \|\nabla f(\boldsymbol{w}_t;\xi'_t) - \nabla f(\boldsymbol{x}_t;\xi'_t)\| + \|\nabla f(\boldsymbol{x}_t;\xi'_t) - \nabla f(\boldsymbol{x}_t;\xi_t)\| \le L_t\eta_t + \|\nabla f(\boldsymbol{x}_t;\xi'_t) - \nabla f(\boldsymbol{x}_t;\xi_t)\|$. So we can further bound the summand in Eq. 20 by

$$\begin{aligned}
&\frac{(2\max\{L_t, M\}\eta_t + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_t\eta_t + \|\nabla f(\boldsymbol{x}_t;\xi'_t) - \nabla f(\boldsymbol{x}_t;\xi_t)\|)^2}{2\delta + 2\sum_{s=0}^{t}(\max\{L_s, M\} + \frac{8(1-\alpha)\tilde{L}_s}{3\alpha})} \\
\le\ &\frac{\eta_t^2(2\max\{L_t, M\} + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_t)^2 + \|\nabla f(\boldsymbol{x}_t;\xi'_t) - \nabla f(\boldsymbol{x}_t;\xi_t)\|^2}{\delta + \sum_{s=0}^{t}(\max\{L_s, M\} + \frac{8(1-\alpha)\tilde{L}_s}{3\alpha})}.
\end{aligned}$$

In the following, we will upper bound the two sums

$$\sum_{t=0}^{T-1} \frac{\eta_t^2(2\max\{L_t, M\} + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_t)^2}{\delta + \sum_{s=0}^{t}(\max\{L_s, M\} + \frac{8(1-\alpha)\tilde{L}_s}{3\alpha})} \quad \text{and} \quad \sum_{t=0}^{T-1} \frac{\|\nabla f(\boldsymbol{x}_t; \xi_t') - \nabla f(\boldsymbol{x}_t; \xi_t)\|^2}{\delta + \sum_{s=0}^{t}(\max\{L_s, M\} + \frac{8(1-\alpha)\tilde{L}_s}{3\alpha})}$$

separately.

By our assumption, $\max\{L_t, \tilde{L}_t\} \leq L^{\max}$ with probability one and $\eta_t \leq \eta^{\max}$. Thus, for all t, we can bound

$$\begin{aligned}
\eta_t^2\Big(2\max\{L_t, M\} + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_t\Big)^2 &= 2\eta_t^2\Big(2\max\{L_t, M\} + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_t\Big)\Big(\max\{L_t, M\} + \frac{4(1-\alpha)}{3\alpha}\tilde{L}_t\Big) \\
&\leq 2(\eta^{\max})^2\frac{6\alpha + 8(1-\alpha)}{3\alpha}\max\{L^{\max}, M\}\Big(\max\{L_t, M\} + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_t\Big) \\
&\leq \frac{16(\eta^{\max})^2\max\{L^{\max}, M\}}{3\alpha}\Big(\max\{L_t, M\} + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_t\Big).
\end{aligned}$$

This further leads to

$$\sum_{t=0}^{T-1} \frac{\eta_t^2(2\max\{L_t, M\} + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_t)^2}{\delta + \sum_{s=0}^{t}(\max\{L_s, M\} + \frac{8(1-\alpha)\tilde{L}_s}{3\alpha})} \tag{21}$$

$$\leq \frac{16(\eta^{\max})^2\max\{L^{\max}, M\}}{3\alpha}\sum_{t=0}^{T-1} \frac{\max\{L_t, M\} + \frac{8(1-\alpha)\tilde{L}_t}{3\alpha}}{\delta + \sum_{s=0}^{t}(\max\{L_s, M\} + \frac{8(1-\alpha)\tilde{L}_s}{3\alpha})}, \tag{22}$$

Now we can apply the following lemma.

Lemma D.1. *For any nonnegative sequence $\{a_t\}_{t=0}^{T-1}$ and $\delta > 0$, it holds that $\sum_{t=0}^{T-1} \frac{a_t}{\delta + \sum_{s=0}^{t} a_s} \leq \log\Big(1 + \frac{\sum_{t=0}^{T-1} a_t}{\delta}\Big)$.*

Proof. For any $t \geq 0$, we have $\frac{a_t}{\delta + \sum_{s=0}^{t} a_s} = 1 - \frac{\delta + \sum_{s=0}^{t-1} a_s}{\delta + \sum_{s=0}^{t} a_s} \leq \log(\frac{\delta + \sum_{s=0}^{t} a_s}{\delta + \sum_{s=0}^{t-1} a_s})$, where we used the fact that $1 - x \leq \log(\frac{1}{x})$ for any $x \geq 0$. Hence, by summing the inequality from $t = 0$ to $t = T - 1$ we obtain $\sum_{t=0}^{T-1} \frac{a_t}{\delta + \sum_{s=0}^{t} a_s} \leq \log(\frac{\delta + \sum_{t=0}^{T-1} a_t}{\delta}) = \log(1 + \frac{\sum_{t=0}^{T-1} a_t}{\delta})$. □

Hence, by applying Lemma D.1 to Eq. 21, we get

$$\begin{aligned}
&\sum_{t=0}^{T-1} \frac{\eta_t^2(2\max\{L_t, M\} + \frac{8(1-\alpha)}{3\alpha}\tilde{L}_t)^2}{\delta + \sum_{s=0}^{t}(\max\{L_s, M\} + \frac{8(1-\alpha)\tilde{L}_s}{3\alpha})} \\
&\leq \frac{16(\eta^{\max})^2\max\{L^{\max}, M\}}{3\alpha}\log\Big(1 + \frac{\sum_{t=0}^{T-1}(\max\{L_t, M\} + \frac{8(1-\alpha)\tilde{L}_t}{3\alpha})}{\delta}\Big) \\
&\leq \frac{16(\eta^{\max})^2\max\{L^{\max}, M\}}{3\alpha}\log\Big(1 + \frac{11\max\{L^{\max}, M\}}{3\alpha\delta}T\Big).
\end{aligned}$$

By our choice of $\eta^{\max} = \sqrt{\alpha}\bar{\eta}$, it becomes $\mathcal{O}\left(\bar{\eta}^2\max\{L^{\max}, M\}\log\left(1 + \frac{\max\{L^{\max}, M\}}{\alpha\delta}T\right)\right)$. For the second term, since $\frac{1}{t+1}\sum_{s=0}^{t}\max\{L_s, M\} \geq M$, using Assumption 2.1, we have

$$\begin{aligned}
\mathbb{E}\Big[\sum_{t=0}^{T-1} \frac{\|\nabla f(\boldsymbol{x}_t; \xi_t') - \nabla f(\boldsymbol{x}_t; \xi_t)\|^2}{\delta + \sum_{s=0}^{t}(\max\{L_s, M\} + \frac{8(1-\alpha)\tilde{L}_s}{3\alpha})}\Big] &\leq \sum_{t=0}^{T-1} \frac{\mathbb{E}[\|\nabla f(\boldsymbol{x}_t; \xi_t') - \nabla f(\boldsymbol{x}_t; \xi_t)\|^2]}{M(t+1)} \\
&= \sum_{t=0}^{T-1} \frac{2\sigma^2}{M(t+1)} \leq \frac{2\sigma^2}{M}(1 + \log(T)).
\end{aligned}$$

Lemma 4.3 now follows from combining the above two bounds.

E ADDITIONAL EXPERIMENTS AND DETAILS

In this section, we discuss additional implementation details, present the seeded runs for the experiments in the main text, and include additional results on two other datasets from torchvision.

E.1 IMPLEMENTATION DETAILS

Hardware Our experiments were conducted on a cluster with NVIDIA A100 GPUs (96GB memory) and 120GB system RAM. The ViT-Tiny runtimes in Table 1 were measured on NVIDIA H100 GPUs. The CIFAR-10 experiments with multiple random seeds required approximately 96 GPU hours, and both the CIFAR-100 and the Flower102 experiments required approximately 192 GPU hours.

Additional hyperparameters Since our main focus is on adaptive learning rate selection, we set all other hyperparameters to their standard default values. Specifically, for ADAM-GALA, we fix $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\delta = 10^{-8}$, consistent with the settings used for Adam and AdamW. For SGD with momentum, we set the momentum parameter to 0.9. While training ViT-Tiny, we set the weight decay for AdamW and SGD to be 0.05, and we kept it 0 for SGD-GALA and ADAM-GALA.

Resnet-18 experiments use a constant learning rate schedule for all algorithms. We observe that a popular schedule, such as cosine, does not change the relative behavior and hence we report the results for a constant learning rate schedule. For the ViT-Tiny experiment, we follow the standard in the literature and run AdamW and SGD with cosine schedules, while we used a constant schedule for our GALA variants to demonstrate the learning rate adaptation without external intervention.

Mechanic The Mechanic algorithm, proposed in Cutkosky et al. (2023a), provides a general framework for adaptively selecting the learning rate of any base optimizer. At each iteration, it proceeds as follows:

- Sample $\xi_t \sim \mathcal{D}$ and compute the stochastic gradient $\boldsymbol{g}_t = \nabla f(\boldsymbol{x}_t; \xi_t)$;
- Use $\boldsymbol{g}_t$ to compute the update direction $\boldsymbol{u}_t$ via the base optimizer and update the cumulative direction $\boldsymbol{\Delta}_{t+1} = \boldsymbol{\Delta}_t + \boldsymbol{u}_t$;
- An internal online learner selects a learning rate s_{t+1};
- Update the iterate: $\boldsymbol{x}_{t+1} = \boldsymbol{x}_1 + s_{t+1}\boldsymbol{\Delta}_{t+1}$.

For example, if the base optimizer is SGD, then $\boldsymbol{u}_t = -\eta \boldsymbol{g}_t$. In our experiments, we apply Mechanic to both SGD and its momentum variant and vary the initial learning rate η, using the official implementation available at `https://github.com/optimizedlearning/mechanic`. Since the performance of the standard and momentum versions was comparable, we only include the Mechanic on SGD without momentum in order to preserve the readability of the plots.

AdGD The update rule of AdGD in Malitsky & Mishchenko (2020) for the stochastic setting is given by

$$\begin{aligned} \eta_t &= \min\left\{\sqrt{1+\alpha\frac{\eta_{t-1}}{\eta_{t-2}}}\eta_{t-1}, \frac{\|\boldsymbol{x}_t - \boldsymbol{x}_{t-1}\|}{2\|\nabla f(\boldsymbol{x}_t;\xi_t) - \nabla f(\boldsymbol{x}_{t-1};\xi_t)\|}\right\}, \\ \boldsymbol{x}_{t+1} &= \boldsymbol{x}_t - \eta_t \nabla f(\boldsymbol{x}_t;\xi_t), \end{aligned} \tag{23}$$

where $\alpha = 1$ in the original algorithm, which is analyzed under deterministic gradients. In practice, the authors recommend using smaller values of α to improve stability and avoid spikes in the loss curve. For example, they report that for ResNet-18 on CIFAR-10, setting $\alpha = 0.02$ yields the best performance. Following their recommendation, we use this value in all of our experiments.

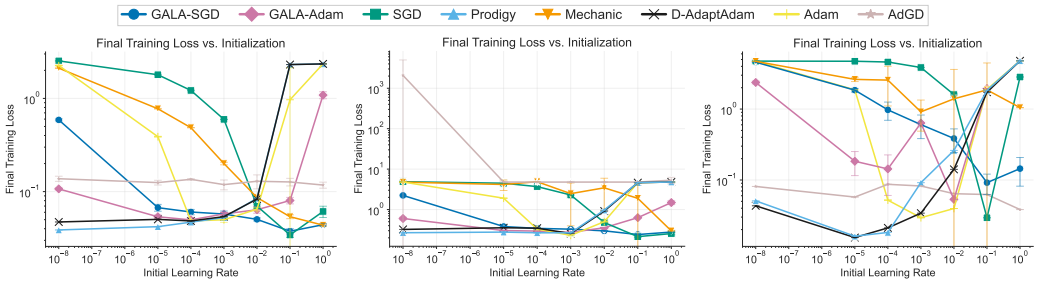
D-Adaptation and Prodigy Among the class of *parameter-free* methods, D-Adaptation (Defazio & Mishchenko, 2023) and Prodigy (Mishchenko & Defazio, 2024) are considered to be state-of-the-art with 3 million downloads each on GitHub. The goal of parameter-free optimization is to eliminate the dependence on the initial learning rate by approximating the initial distance to the solution. Essentially, the step size is based on the AdaGrad step size with a *certificate* for initial distance, i.e., $\|x_0 - x^*\|$, that iteratively improves every step. Note that the algorithm still requires an initial value for the certificate, d_0, which is equivalent to the initial learning rate. We keep all the other parameters of both algorithms

the same as in the GitHub implementation and the paper. We downloaded and installed the official algorithm packages from `https://github.com/facebookresearch/dadaptation` for D-Adaptation and `https://github.com/konstmish/prodigy` for Prodigy.

E.2 Experiments

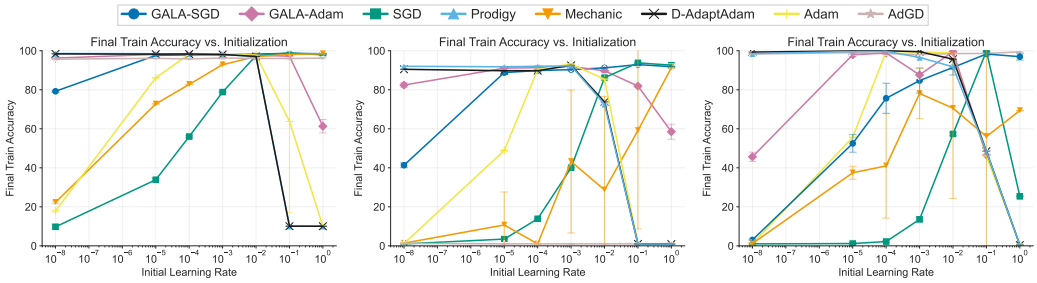
E.2.1 ResNet-18

Alongside CIFAR-10, we ran experiments with two more datasets, CIFAR-100 Krizhevsky (2009) and Oxford 102 Flower dataset (Flower102) Nilsback & Zisserman (2008), on ResNet-18. In order to quantify the error due to randomness, we ran the experiments with three different random seeds, which we report as error bars. Figure 5, Figure 6, and Figure 7 report the final training loss, training accuracy, and testing accuracy with respect to different learning rates, respectively.



(a) CIFAR-10 (b) CIFAR-100 (c) Flowers102

Figure 5: **Final training loss** obtained from different initial learning rates for CIFAR-10, CIFAR-100, and Flower102. We compare the performance of SGD-GALA, Adam-GALA against SGD, Adam, AdGD, Mechanic, D-Adaptation(Adam) and Prodigy. We initialize each algorithm with learning rates $[1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-8}]$ and execute 3 seeded runs.

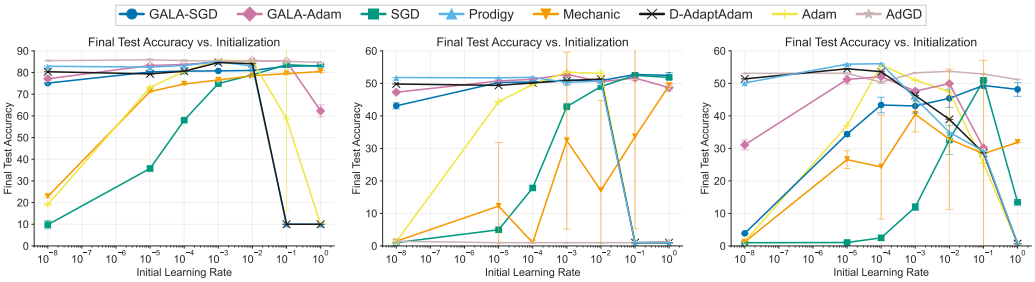


(a) CIFAR-10 (b) CIFAR-100 (c) Flowers102

Figure 6: **Final training accuracy** obtained from different initial learning rates for CIFAR-10, CIFAR-100, and Flower102. We compare the performance of SGD-GALA, Adam-GALA against SGD, Adam, AdGD, Mechanic, D-Adaptation(Adam) and Prodigy. We initialize each algorithm with learning rates $[1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-8}]$ and execute 3 seeded runs.

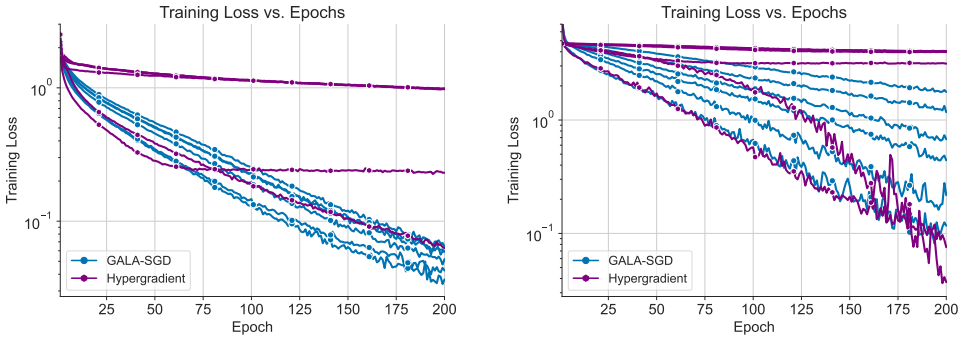
Across the three different datasets, we observe that our method, SGD-GALA and Adam-GALA, remains robust with respect to the initial learning rates and has negligible variance due to random seeds. SGD and Adam are sensitive to the choice of the initial learning rate; a small learning rate prevents progress while relatively large values yield unpredictable behavior with abrupt changes in performance.

The results on all three datasets (although more pronounced for CIFAR-10 and CIFAR-100) show that Mechanic tends to perform better with larger learning rates, but the variance is high with different random seeds. Interestingly, AdGD fails on the CIFAR-100 dataset; as we will discuss later in more detail when analyzing the learning rate evolution of the method, this is likely due to the fact that



(a) CIFAR-10 (b) CIFAR-100 (c) Flowers102

Figure 7: **Final test accuracy** obtained from different initial learning rates for CIFAR-10, CIFAR-100, and Flower102. We compare the performance of SGD-GALA, ADAM-GALA against SGD, Adam, AdGD, Mechanic, D-Adaptation(Adam) and Prodigy. We initialize each algorithm with learning rates $[1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-8}]$ and execute 3 seeded runs.



(a) **CIFAR-10** (b) **Flower-102**

Figure 8: Comparison of **training loss** between SGD-GALA and hypergradient descent from different initial learning rates.

its learning rate becomes too large in some scenarios. For other datasets, AdGD shows consistent convergence across different initialization with great test performance and slightly worse training loss/accuracy. The behavior of parameter-free methods is quite close to each other for all three datasets. Both Prodigy and D-Adaptation show one of the best performances for training and testing when the initial step size is small. However, the performance drops sharply when the initialization is larger than 10^{-2}.

Compared to Mechanic and AdGD, the variance for different seeds is smaller for SGD-GALA and ADAM-GALA. Among the GALA-variants, SGD-GALA performs better than ADAM-GALA for larger learning rates.

E.2.2 COMPARISON WITH HYPERGRADIENT DESCENT

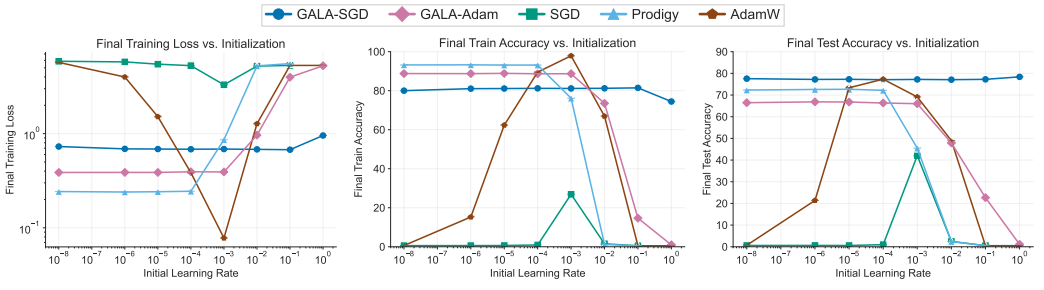
Although HGD only comes with theoretical guarantees in deterministic convex settings, we include it here for comparison due to its relevance. Unlike our method, HGD introduces an additional hyperparameter, *hypergradient step size*, to update the *actual learning rate* η_t. We trained ResNet-18 on the CIFAR-10 and Flower-102 datasets, selecting the initial learning rate η_0 and the hypergradient step size γ_0 from the set $\{10^{-i} : i = 0, \ldots, 5\}$, resulting in 36 runs per dataset. On both datasets, HGD diverged for all initial learning rates when the hypergradient step size was large ($\gamma_0 > 0.1$). While smaller values of γ_0 led to more stable behavior, they also slowed the evolution of η_t, often causing the decrease in training loss to stagnate. Hence, for HGD, we pick the best-performing hypergradient step size for each dataset and compare it with our method across all initial values of η_0 in Figure 8.

E.2.3 ViT on Tiny-ImageNet-200

The robustness provided by GALA framework is better displayed with the ViT experiments; SGD-GALA improves the performance of SGD from all initializations and shows almost the same performance for all initializations. Adam-GALA shows a similar performance improvement with respect to AdamW, particularly for the initialization regimes where it performs poorly, which corresponds to small or large initial values of the learning rate. However, we underline that the best-performing AdamW is better than that of Adam-GALA.

During training, the best-performing configuration for AdamW is better than that of any other method we tested, for which we use a prescribed set of parameters and such a behavior is expected. However, SGD-GALA has the best test accuracy, matching and even surpassing the test performance of AdamW. An interesting observation is that SGD-GALA achieves this performance from any initial learning rate value, which validates our claims on stability and robustness to initialization.

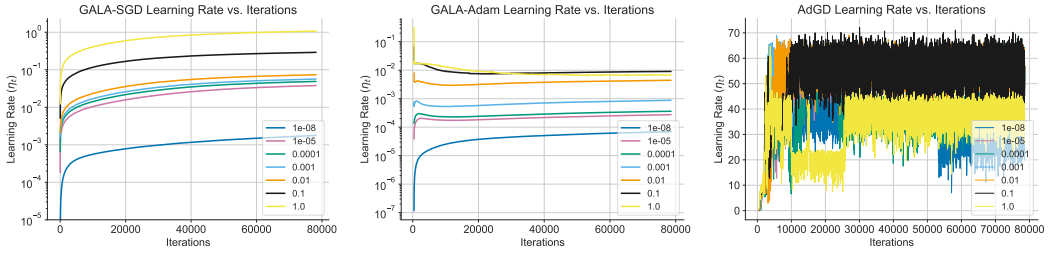
In all our experiments, GALA provides a trade-off between training performance and robustness to initialization; for some experiments performance matches the best-performing algorithm, while for all scenarios, GALA variants of the standalone method show robustness to variation in initial learning rate. Prodigy has strong performance from small initializations across the board, but a sharp degradation happens for learning rates larger than 10^{-3}.



(a) Final Training Loss (b) Final Training Accuracy (c) Final Test Accuracy

Figure 9: **ViT (Tiny variant) on Tiny-ImageNet dataset.** We compare SGD-GALA and Adam-GALA against SGD, AdamW and Prodigy. Each algorithm is initialized with learning rates $[1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-8}]$ and other parameters are set to default values. The runs are averaged over 2 random seeds.

E.2.4 Learning rate evolution



(a) SGD-GALA (b) Adam-GALA (c) AdGD

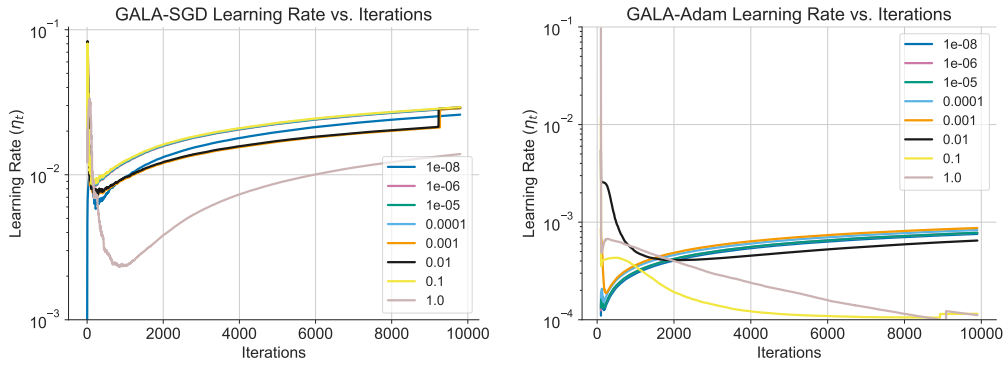
Figure 10: Comparison of **learning rate evolution** for SGD-GALA, Adam-GALA and AdGD on the CIFAR-100 dataset, averaged over 3 runs.

To better understand the convergence behavior of our method, we visualize the learning rate dynamics during training. We demonstrate the evolution of learning rates for SGD-GALA and Adam-GALA for different models and datasets.

As shown in Figure 10a, the learning rate of SGD-GALA evolves similarly and converges to similar values across a wide range of initialization, excluding extreme cases such as $\eta = 1, 0.1$, or 10^{-8}. This

convergence likely explains the robustness of SGD-GALA to the choice of initial learning rate. Also, in Figure 10b, we observe that, depending on the initial value, the learning rate of ADAM-GALA adapts over time and can both increase and decrease, consistent with the trend seen in Figure 4 on CIFAR-10. In most cases, the learning rate stabilizes between 10^{-2} and 10^{-3}, which roughly corresponds to the best fixed learning rate for Adam according to Figure 7b. By contrast, Figure 10c shows that the learning rate chosen by AdGD tends to oscillate and frequently becomes excessively large, which may contribute to its degraded performance. While AdGD performs competitively on CIFAR-10, its behavior on CIFAR-100 suggests that it may be less robust and that the hyperparameter α in Eq. 23 may require retuning for stable performance on new datasets.

Figure 11 shows the learning rate evolution for ViT + Tiny-ImageNet for SGD-GALA and ADAM-GALA. For this setup, the learning rate evolution across different initializations demonstrates a more consistent behavior. Specifically, SGD-GALA converges to the same narrow value range for the learning rate from different initializations between 1 and 10^{-8}. While doing so, the learning rate decreases initially and starts increasing for most of the execution. For ADAM-GALA, the behavior seems a bit more complex; the learning rate fluctuates for a while until it attains a monotonic, increasing trend. Note that the learning rate for initializations between $[10^{-2}, 10^{-8}]$ converges to almost the same value with an increasing behavior for most of the run, while from initial values of 1 and 0.1, the learning rate tends to decrease, which hurts the performance as shown in the convergence plots.



(a) SGD-GALA (b) ADAM-GALA

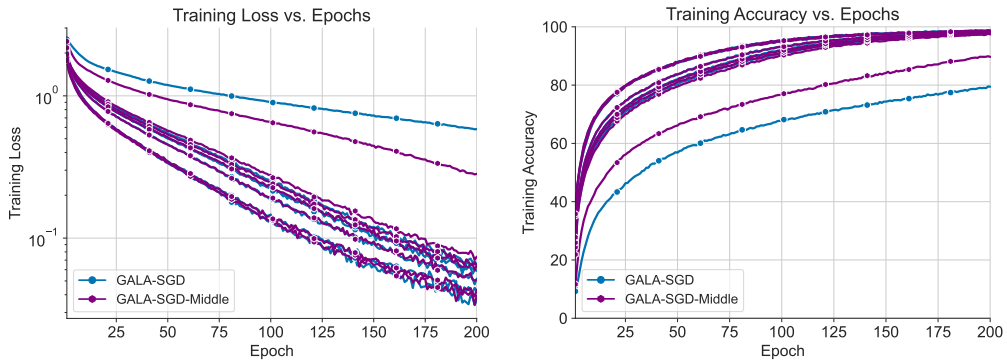
Figure 11: Comparison of **learning rate evolution** for SGD-GALA and ADAM-GALA for training ViT-Tiny on Tiny-ImageNet, averaged over 2 runs.

E.2.5 PRACTICAL CONSIDERATIONS

We have discussed in Remark 3.3 and the implementation details in Section 5 that we propose principled modifications for the implementation of our algorithm to improve computational efficiency. One such modification concerns the computation of stochastic gradient $\nabla f(\boldsymbol{w}_t; \xi_t')$ at a random middle point $\boldsymbol{w}_t = \boldsymbol{x}_t + s_t(\boldsymbol{x}_{t+1} - \boldsymbol{x}_t)$ along the segment $\boldsymbol{x}_{t+1} - \boldsymbol{x}_t$, where $s_t \sim \text{Uniform}[0, 1]$. Compatible with our analysis, we set $\xi_t' = \xi_{t+1}$ and also pick a deterministic value $s_t = 1$, which gives us $\nabla f(\boldsymbol{w}_t; \xi_t') \to \nabla f(\boldsymbol{x}_{t+1}, \xi_{t+1})$. This is a crucial modification that enables us to reuse $\nabla f(\boldsymbol{x}_{t+1}; \xi_{t+1})$ for the next iteration. Moreover, this modification has a negligible effect in the convergence performance. To verify, we compare two variants of implementation in Figure 12; GALA-SGD is the implementation as described in Eq. Eq. 10 and GALA-SGD-MIDDLE denotes the version that replaces $\nabla f(\boldsymbol{x}_{t+1}; \xi_{t+1})$ with $\nabla f(\boldsymbol{w}_t; \xi_{t+1})$. As demonstrated in Figure 12, the practical choice of $\nabla f(\boldsymbol{x}_{t+1}; \xi_{t+1})$ instead of the theoretical prescription $\nabla f(\boldsymbol{w}_t; \xi_{t+1})$ yields almost identical performance from different initial learning rates. One efficiency advantage of approximating $\boldsymbol{w}_t$ with $\boldsymbol{x}_{t+1}$ is that during runtime, we end up changing the model parameters less often. Hence, the proposed, practical version of GALA-SGD in Eq. Eq. 10 enjoys improved runtime performance.

E.2.6 RUNTIME COMPARISON

Since GALA requires two stochastic gradient evaluations per iteration, it incurs a higher per-iteration cost than standard SGD and Adam. As a representative example, we report in Table 1 the average



(a) Training Loss

(b) Training Accuracy

Figure 12: Comparison of our algorithm implemented as in Eq. Eq. 10 (SGD-GALA) and the version that uses the gradient at a random middle point $\boldsymbol{w}_t$ (GALA-SGD-MIDDLE) for training ResNet-18 on CIFAR-10 from different initial learning rates selected from the range $[10^{-8}, 1]$.

Table 1: Average runtime comparison. On the left, SGD vs. GALA-SGD for training ResNet-18 on CIFAR-10 for 200 epochs. On the right, AdamW vs GALA-Adam for training ViT-Tiny on Tiny-ImageNet for 100 epochs (all runs and seeds included). Each entry reports the mean wall-clock time in seconds, with the standard deviation in parentheses.

Algorithm	Runtime for **Resnet-18**
SGD	1935.4 (71.3)
GALA-SGD	3011.4 (99.8)

Algorithm	Runtime for **ViT-Tiny**
AdamW	6223.2 (90)
GALA-Adam	9423 (649.8)

wall-clock times for SGD-GALA and ADAM-GALA compared to SGD and AdamW, respectively. As reported in Section 5, it takes 200 epochs to train ResNet-18 on CIFAR-10, and 100 epochs to train ViT-Tiny on Tiny-ImageNet. Standalone algorithms and their GALA version are trained on identical hardware. We observe that SGD-GALA introduces an approximately 55.6% runtime overhead compared to SGD for Resnet-18 experiments, while ADAM-GALA incurs a similar 51.4% overhead for training the ViT-Tiny model. That said, this additional cost is often offset by the substantially reduced tuning effort, as GALA exhibits robust performance across a wide range of learning-rate initializations.